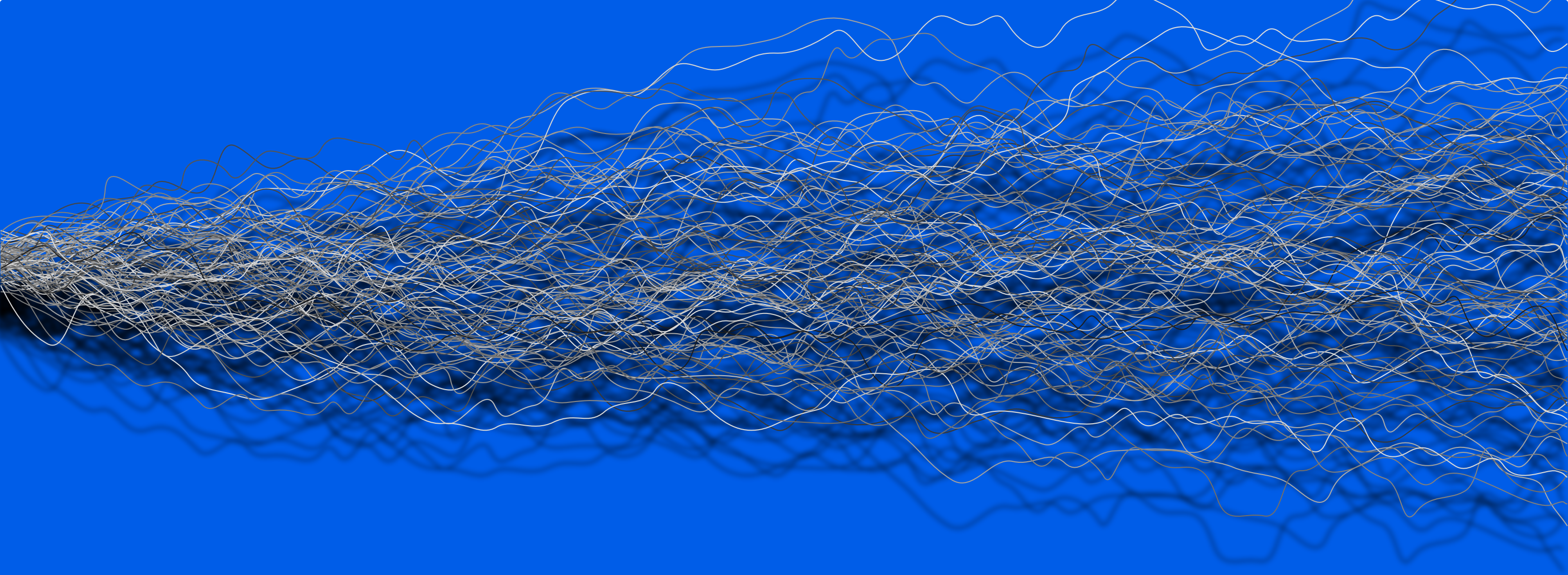


Modeling Biotechnological Processes under Uncertainty

Anaerobic Digestion as Case Study

Živko Južnič-Zonta

Oficina de Doctorat
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Curs acadèmic: 2011/2012

Acta de qualificació de tesi doctoral

Nom i cognoms
Živko Južnič-Zonta

DNI / NIE / Passaport
Y0093083Q

Programa de doctorat
Enginyeria Ambiental

Unitat estructural responsable del programa
Institut Universitari de Recerca en Ciència i Tecnologies de la Sostenibilitat

Resolució del Tribunal

Reunit el Tribunal designat a l'efecte, el doctorand / la doctoranda exposa el tema de la seva tesi doctoral titulada

Modeling Biotechnological Processes under Uncertainty. Anaerobic Digestion as Case Study

____.

Acabada la lectura i després de donar resposta a les qüestions formulades pels membres titulars del tribunal, aquest atorga la qualificació:

□ APTA/E □ NO APTA/E

(Nom, cognoms i signatura) President/a	(Nom, cognoms i signatura) Secretari/ària

(Nom, cognoms i signatura) Vocal	(Nom, cognoms i signatura) Vocal	(Nom, cognoms i signatura) Vocal

______________________, _______ d'/de __________________ de _______________

El resultat de l'escrutini dels vots emesos pels membres titulars del tribunal, efectuat per l'Escola de Doctorat, a instància de la Comissió de Doctorat de la UPC, atorga la MENCIÓ CUM LAUDE:

□ SI □ NO

(Nom, cognoms i signatura) Presidenta de la Comissió de Doctorat	(Nom, cognoms i signatura) Secretària de la Comissió de Doctorat

Barcelona, _______ d'/de __________________ de _______________

PhD Thesis

UPC - Program on Environmental Engineering

Modeling Biotechnological Processes under Uncertainty

Anaerobic Digestion as Case Study

Živko Južnič-Zonta

July 2012, Barcelona

Universitat Politècnica de Catalunya
Department of Agrifood Engineering and Biotechnology
GIRO Joint Research Unit IRTA-UPC

Supervisor

Prof. Dr. Xavier Flotats i Ripoll

Co-supervior

Dr. Albert Magrí Aloy

Acknowledgments

I would like to thank Xavier Flotats i Ripoll and Albert Magrí Aloy for their advice throughout my PhD; Belén Fernández, Laura Tey and Angela Rodríguez-Abalde for assisting me in acquiring laboratory skills and for their frutful discussions. I would like to thank my co-authors in the submitted publications leading up to my thesis: Jordi Palatsi, Albert Magrí Aloy, Darko Vrečko, Juš Kocijan, Maria Magdalena Alves, and Xavier Flotats i Ripoll. I thank Darko Vrečko for making possible my stage at Department of Systems and Control, “Jožef Stefan” Institute, and for providing an exiting working atmosphere. I would like to thank Universitat Politècnica de Catalunya and GIRO Joint Research Unit IRTA-UPC (formerly GIRO Technological Centre), which funded my studentship.

Summary

In engineering practice, when an explicit model of a system is available, numerical experiments can be performed in order to predict the future behavior of the system, explain or describe its hidden state, guide data collection, etc. Typically, the dynamics of the system are complex and difficult to observe with precision. Any approximation of the observed reality within an explicit model necessary implies uncertainty, which should be characterized and quantified to build confidence over model results. Uncertainty associated with model-parameter and its implications for bio-process optimization are of main concern in this PhD work.

As a bio-process case study, the anaerobic digestion is considered for modeling. The production of biogas by controlled anaerobic digestion could be a profitable activity, apart of being a renewable energy source. However, the margins to improve this technology are wide. Anaerobic co-digestion with two or more input materials is a way to make low biogas yield biomass applicable at industrial scale. Among the possible co-substrates, lipids-rich wastes are attractive for their high energetic potential. The main limiting factor for this strategy is the inhibition of anaerobic digestion by long chain fatty acids. Modeling provides a useful approximation of the complex and delicate microbiology activity of this anaerobic digestion system.

The underlying goal of the PhD project is to improve biotechnological processes with the aid of modeling and uncertainty analysis. With this goal in mind, a general purpose, user-friendly, simulation environment called “virtual plant” (VP) was build and applied to anaerobic co-digestion and activated sludge modeling. Within the VP tool, new core dynamics of the long chain fatty acids (LCFA) inhibition process were proposed and tested and different inferential procedures for the estimation of parameter-uncertainty were compared. Finally, a proposed multi-criteria analysis under uncertainty and multiplicity was applied to an industrial anaerobic co-digestion biogas plant.

In conclusion, the developed VP toolkit was found reliable and user-friendly when modeling activated sludge and anaerobic digestion systems. The proposed LCFA-inhibition model was able to reproduce correctly the experimental data at hand and enabled its interpretation. However, uncertainty estimation of parameters and falsification of the

proposed model of LCFA-inhibition are still missing. The Bayesian procedure was proved useful when addressing the estimation of parameter uncertainty of anaerobic digestion and activated sludge models. A considerable improvement in the operation efficiency and reliability of an industrial biogas plant was possible within the proposed multi-criteria analysis. However, future work is needed to improve the procedure of elicitation of the inputs for this multi-criteria analysis and decrease its computational burden.

Resum

En la pràctica de l'enginyeria, quan un model explícit d'un procés està disponible, es poden realitzar experiments numèrics per tal de predir el comportament futur del sistema, explicar o descriure el seu estat ocult, guiar la recopilació de dades,... Generalment, les dinàmiques del sistema són complexes i difícils d'observar amb precisió. Qualsevol aproximació de la realitat observada per mitjà d'un modelatge implica necessàriament incertesa. Per fomentar la confiança en els resultats del model, aquesta incertesa ha de ser caracteritzada i quantificada de forma explícita. En aquest projecte de tesi, particular atenció es proporciona a la incertesa associada als paràmetres del model i les seves implicacions per a l'optimització de bio-processos.

Com a cas d'estudi, es considera per a la modelització la digestió anaeròbia. La producció controlada de biogàs per digestió anaeròbica s'ha trobat una activitat rendible, a més de ser una font d'energia renovable. No obstant això, els marges de millora per a aquesta tecnologia són amplis. La co-digestió anaeròbia amb dos o més materials d'entrada és una manera de fer que la biomassa de baixa producció de biogàs sigui aplicable a escala industrial. Entre els possibles co-substrats, els residus orgànics rics en lípids resulten atractius pel seu alt potencial energètic. El principal factor limitant per a aquesta estratègia és la inhibició de la digestió anaeròbica pels àcids grassos de cadena llarga. La modelització matemàtica ofereix una aproximació útil de la complexa i delicada activitat microbiologia d'aquest sistema de digestió anaeròbica.

L'objectiu subjacent del projecte de tesi és millorar ells processos biotecnològics amb l'ajuda de la modelització i l'anàlisi d'incertesa. D'acord amb aquest objectiu, es desenvolupa un entorn de simulació anomenat "planta virtual" (VP) amb la fi de aplicar-lo al modelatge de la co-digestió anaeròbia i fangs activats. A l'entorn de la VP, es proposa i testeja noves dinàmiques fonamentals del procés d'inhibició pels àcids grassos de cadena llarga i es compara diferents procediments d'inferència per l'estimació del la incertesa dels paràmetres. D'altra banda, es proposa una anàlisi de criteris múltiples en condicions d'incertesa y multiplicitat d'equilibris. El mètode s'aplica a una planta industrial de co-digestió anaeròbica.

Com a conclusió, l'eina de la "planta virtual" es va trobar fiable i fàcil d'usar en el

modelat dels processos de tractament biològics com fangs activats i digestió anaeròbia. El model d'inhibició per àcids grassos a cadena llarga ha estat capaç de reproduir i ha consentit de interpretar les dades experimentals obtingudes en prèvies investigacions. No obstant això, l'estimació de la incertesa dels paràmetres i la falsificació del model d'inhibició són tasques d'investigació futura. El procediment d'inferència Bayesiana s'ha demostrat útil per enfrontar-se amb èxit al problema de l'estimació de la incertesa dels paràmetres relatius a models de la digestió anaeròbia i dels fangs activats. La anàlisi de criteris múltiples ha permès una considerable millora en l'eficiència i de la fiabilitat d'operació d'una planta industrial de biogàs. No obstant això, com a treball futur es fa necessari millorar el procediment d'obtenció de les entrades a l'anàlisi de criteris múltiples i disminuir la càrrega computacional requerida per aquesta anàlisi.

Resumen

En la práctica de la ingeniería, cuando un modelo explícito de un proceso está disponible, se pueden realizar experimentos numéricos para predecir el comportamiento futuro del sistema, explicar o describir su estado oculto, guiar la recopilación de datos,... Generalmente, las dinámicas del sistema son complejas y difíciles de observar con precisión. Cualquier aproximación de la realidad observada a través de un modelado implica necesariamente incertidumbre. Para fomentar la confianza en los resultados del modelo, esta incertidumbre debe ser caracterizada y cuantificada de forma explícita. En este proyecto de tesis, particular atención se proporciona a la incertidumbre asociada a los parámetros del modelo y sus implicaciones para la optimización de bio-procesos.

Como caso de estudio, se considera para la modelización la digestión anaerobia. La producción controlada de biogás por digestión anaeróbica se ha encontrado una actividad rentable, además de ser una fuente de energía renovable. Sin embargo, los márgenes de mejora para esta tecnología son amplios. La co-digestión anaerobia con dos o más materiales de entrada es una manera de hacer que la biomasa de baja producción de biogás sea aplicable a escala industrial. Entre los posibles co-sustratos, los residuos orgánicos ricos en lípidos resultan atractivos por su alto potencial energético. El principal factor limitante para esta estrategia es la inhibición de la digestión anaeróbica por los ácidos grasos de cadena larga. La modelización matemática ofrece una aproximación útil de la compleja y delicada actividad microbiológica de este sistema de digestión anaeróbica.

El objetivo subyacente del proyecto de tesis es mejorar los procesos biotecnológicos con la ayuda de la modelización y el análisis de incertidumbre. De acuerdo con este objetivo, se desarrolla un entorno de simulación llamado “planta virtual” (VP) con el fin de aplicarlo al modelado de la co-digestión anaerobia y fangos activados. En el entorno de la VP, se propone y testea nuevas dinámicas fundamentales del proceso de inhibición por ácidos grasos de cadena larga y se compara diferentes procedimientos de inferencia para la estimación del la incertidumbre de los parámetros. Por otra parte, se propone un análisis de criterios múltiples en condiciones de incertidumbre y multiplicidad de equilibrios. El método se aplica a una planta industrial de co-digestión anaeróbica.

Como conclusión, la herramienta de la “planta virtual” se encontró fiable y fácil de usar en el modelado de los procesos de tratamiento biológicos como lodos activados y digestión anaerobia. El modelo de inhibición por ácidos grasos a cadena larga ha sido capaz de reproducir y ha permitido de interpretar los datos experimentales obtenidos en previas investigaciones. Sin embargo, la estimación de la incertidumbre de los parámetros y la falsificación del modelo de inhibición son tareas de investigación futura. El procedimiento de inferencia Bayesiana se ha demostrado útil para enfrentarse con éxito al problema de la estimación de la incertidumbre de los parámetros relativos a modelos de la digestión anaerobia y de los lodos activados. La propuesta análisis de criterios múltiples ha permitido una considerable mejora en la eficiencia y de la fiabilidad de operación de una planta industrial de biogás. Sin embargo, como trabajo futuro se rende necesario mejorar el procedimiento de obtención de las entradas al análisis de criterios múltiples y disminuir la carga computacional requerida por tal análisis.

Contents

List of Figures

List of Tables

1 Introduction and Objectives

1.1 Problem Setting

In the last decades, the growing concern for global warming issues and a more stringent environmental legislation has boosted the growth of the renewable energy sector. Biomass usage for biogas production, through anaerobic digestion, is one of the many renewable energy alternatives to reduce CO_2 emissions by substitution of fossil fuels and reducing uncontrolled CH_4 emissions from the same biomass. Apart from the concern for global warming, the production of biogas by controlled anaerobic digestion (AD) could be a profitable activity, since biogas is a valuable product that could be converted to thermal and electrical energy, or simply by injecting it after treatment in the local natural gas network. Moreover, digestate produced during AD can be used as fertilizer or soil conditioner.

Unfortunately, there are many types of biomass that could not be used in AD directly because of their low economic viability at industrial scale. For example, animal manure has a low biogas yield, but on the other hand it has the advantage to be abundant in regions with a dense animal farming activity. Anaerobic co-digestion with two or more input materials is a way to make low biogas yield biomass applicable at real scale. Among the possible co-substrates, lipids-rich wastes are attractive for their high energetic potential in terms of biogas specific production. The drawback of using lipids-rich wastes is the inhibitory, even though reversible, effect on the anaerobic process by their first degradation products, the long chain fatty acids (LCFA).

The co-digestion process emerges from a complex interaction of heterogeneous microorganism populations. Moreover, the reactor environment is of multiphase type, since a solid-liquid-gas interface is present. Those multiphysics processes can be described in a rigorous way by an explicit mathematical model. When this explicit model is coded inside a simulation environment, numerical experiments can be performed in order to predict the future behavior of the system, explain or describe its hidden state, guide data collection, etc.

In the last decade the Anaerobic Digestion Model No.1 (ADM1), developed by Bastone et al. (2002), has been the reference explicit model for AD. ADM1 has already been modified for anaerobic co-digestion modeling (Boubaker and Ridha, 2008; Galí et al., 2009; Zaher et al., 2009; Astals et al., 2011). However, the uncertainty involved in the model results used for the optimization of the co-digestion process has not been explicitly acknowledged so far.

1.2 Anaerobic Digestion (AD)

Anaerobic microbiological digestion is a series of processes in which a heterogeneous population of micro-organisms breakdown biodegradable materials in the absence of oxygen in order to derive energy for their growth with a resulting production of CH_4.

A simplified schematic representation of AD of a general complex organic substrate X_c is given as Figure1.2.1. There are four main bio-process phases that take place:

Hydrolysis. Fermentation bacteria secrets multiple extracellular enzymes in order to breakdown complex suspended compounds and colloidal materials. Reactions catalyzed by cellulase, protease and lipase reduce carbohydrates X_{ch}, proteins X_{pr} and lipids X_{li} into their correspondent monomers of sugars S_{su}, amino acids S_{aa} and long chain fatty acids S_{fa}.

Acidogenesis. Converts the soluble organic compounds produced during hydrolysis to volatile fatty acids (VFA), NH_4^+, H_2 and HCO_3^-. Long chain fatty acids are proceeded via $\beta-$oxidation reactions. During $\beta-$oxidation, two-carbon molecules acetyl-CoA are repeatedly cleaved from the fatty acid. Acetyl-CoA can then enter the TCA cycle, which produces NADH and FADH, which are subsequently used in the electron transport chain to produce ATP.

Acetogenesis. Converts the VFA (mainly butyrate S_{bu}, propionate S_{pro} and valerate S_{va}) to acetate S_{ac} and H_2.

Methanogenesis. Acetate is converted to CH_4 by acetoclastic bacteria, while hydrogenotrophic bacteria produce CH_4 from H_2 and CO_2.

The disintegration process (Figure1.2.1) is considered as a non-biological pre-lysis step, since the complex particulate substrates X_c breakdown into its biodegradable and inert parts simply by phase separation, inter-particle shearing and lysis by thermal or other types of external energy supply. Moreover, the disintegration step serves as a repository for the death biomass, which is recirculated as a co-substrate in the anaerobic bio-process.

During stable digestion, disintegration and hydrolysis are the limiting steps of the process for particulate organic substrates. With "stable" we mean that various biological conversions remain sufficiently coupled during the process in order to prevent the accumulation of intermediate compounds. On the other side, methanogenesis phase is the most sensible to environmental changes in the process. For example, an accumulation of VFA will decrease pH, which will in turn inhibit the methanogenesis process and reduce the consumption of acetate even more.

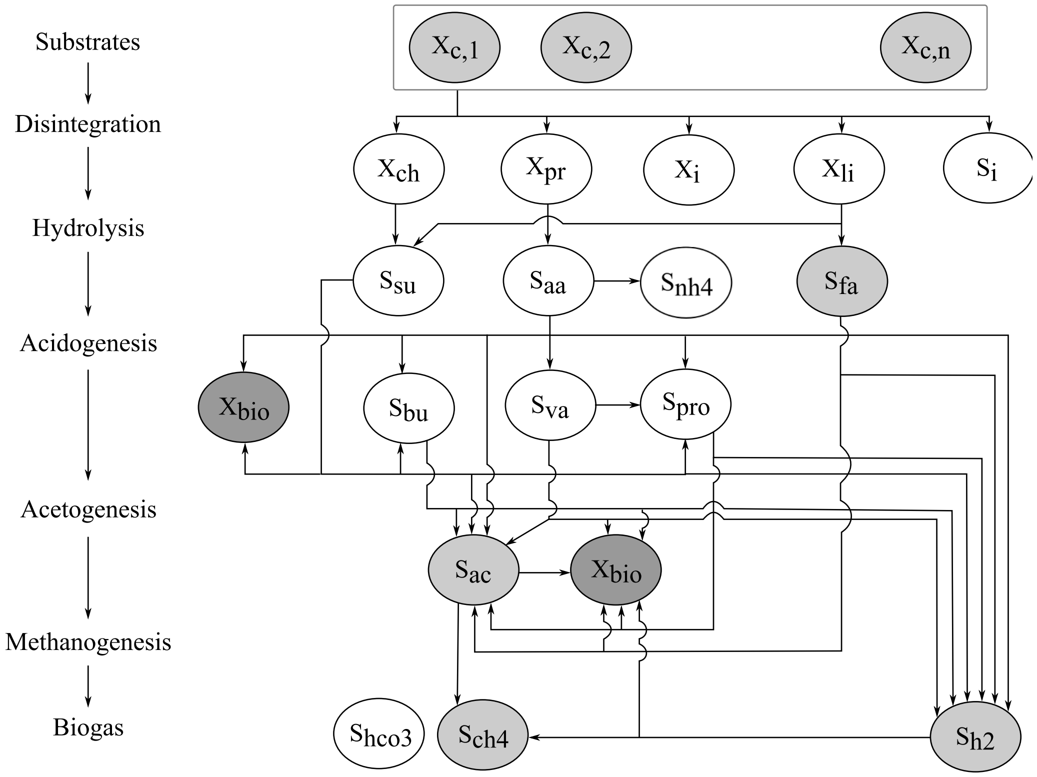


Figure 1.2.1: The main anaerobic process metabolites. Particulate compounds are: decayed biomass (X_{dec}), complex organic substrate (X_c), carbohydrates (X_{ch}), proteins (X_{pro}), inerts (X_i), lipids (X_{li}) and micro-organism biomass (X_{bio}). Soluble compounds are: amino acids (S_{aa}), sugars (S_{su}), long chain fatty acids (S_{fa}), butyrate (S_{bu}), valerate (S_{va}), propionate (S_{pro}), acetate (S_{ac}), methane (S_{ch4}), hydrogen (S_{h2}), inorganic carbon (S_{hco3-}) and nitrogen (S_{nh4+}).

Apart from pH, another environmental factor affects considerably the anaerobic process: temperature determines psychrophilic ($10 - 20\,^{\circ}\mathrm{C}$), mesophilic ($20 - 40\,^{\circ}\mathrm{C}$), or thermophilic $50 - 60\,^{\circ}\mathrm{C}$ digestion . Since industrial plant anaerobic reactors are built with control of temperature in mind, it is kept constant at a relative optimal level in relation to the objective of the treatment. The choice to maintain as constant as possible the temperature is dictated by the fact that the bio-process could slow down if biomass should acclimatize to varying environmental condition. The same applies to the fed substrate composition. Thus, it is important to determine optimal feed substrate mixtures.

Several compounds could exhibit toxic\inhibitory effects: free ammonia ($\mathrm{N}H_3$), sulphide, LCFA, cations (Na^+, K^+ and Ca^{2+} and Mg^{2+}), light and heavy metal ions and xenobiotics. Free ammonia is formed by the protonation of ammonia, which is produced by biological degradation mostly from urea and proteins present in the feed substrate. Ammonia production is especially abundant where wastes from slaughterhouse, fish and dairy industry are fed in co-digestion. Nevertheless, ammonia is needed because it is an essential micro-nutrients.

Methanogenic inhibition from sulfide is present only when sulfate is present in the feed, and is mostly caused by competition for lactate and acetate between acetoclastic and sulfate reducing bacteria. Moreover, sulfide is toxic for various bacteria groups.

Lipids are present in high quantities in the wastes produced by slaughterhouses and meat-processing, dairy, fish-processing, starch-processing, livestock farms, wool scouring facilities and edible and oil processing facilities. For example, in the EU, approximately $17 \cdot 10^6$ tonn/year of slaughter byproducts are produced by the meat industry, where only 50% of the total meat production is for human consumption (Woodgate and van der Veen, 2004). The remaning animal byproducts are a considerable potential resource for anaerobic co-digestion plants. Moreover, there are promising estimations of biogas yields from different lipids-rich substrates. If biomass growth is neglected, the theoretical estimation of the $\mathrm{CH_4}$ yield from lipids ($\mathrm{C_{57}H_{104}O_6}$) is of $1.014\,\mathrm{m^3CH_4/kgVS}$, while for proteins ($\mathrm{C_5H_7ON_5}$) is of $0.496\,\mathrm{m^3CH_4/kgVS}$ and for carbohydrates ($[\mathrm{C_6H_{10}O_5}]_\mathrm{n}$) is of $0.415\,\mathrm{m^3CH_4/kgVS}$ (Angelidaki et al., 1999). The theoretical $\mathrm{CH_4}$ content in biogas is of 70% from lipids, while it is only of 50% for proteins and carbohydrates.

From practice, there has been found for example that slaughterhouse waste reach a biogas yield production of 0.3 to $0.7\,\mathrm{m^3CH_4/kgTS}$, while animal fat could reach $1\,\mathrm{m^3CH_4/kgTS}$ (Hejnfelt and Angelidaki, 2009). Unfortunately, this high lipid-methane-yields come along with a significant inhibitory effect caused by digestion intermediates of LCFA and thus could not reach the high theoretical potentials mentioned above. To avoid LCFA inhibition/toxic effects over AD, lipid rich wastes are fed in co-digestion with, for example, liquid pig manure in order to dilute the eventually accumulated LCFAs in the anaerobic reactor, although this mixture can lead to a high ammonia concentration medium.

1.3 Modeling of Anaerobic Digestion

For waste-water applications, deterministic models like the ASM No.1-3 (Henze et al., 2000) or ADM No.1 (Bastone et al., 2002) are widely used, while stochastic differential equation (SDE) representations or agent-based models that has become very popular during the last decade in the filed of biology and social sciences are somehow left aside. The reason why macroscopic deterministic models are still preferred to SDE or agent-based models, is that they are typicaly as effective as microscopic representations and computationally less expensive. Nevertheless, stochastic models offer a unique insights over the micro-scale, which fosters the interpretation of basic theory of the system process (Picioreanu et al., 2005). Moreover, Gujer (2002) has shown that one of the main drawbacks of deterministic models is that they are only system specific: the estimated lumped kinetic parameters can only be valid for specific flow schemes. On the other hand, microscopic models are applicable to any flow schema.

In recent years, there has been a considerable effort of the AD community to extend the applicability of the standard ADM1 model, as reported by Appels et al. (2008) and references therein. There are two main reasons why the detail of description of the biochemical processes considered by ADM1 has been constantly rising. The first is that a better substrate characterization has become available and because of the need to apply ADM1 for anaerobic co-digestion simulations, e. g. agro-waste (Galí et al., 2009), slaughterhouse solid waste (López and Borzacconi, 2010) or cattle manure and renewable energy crops (Lübken et al., 2007) AD-applications. The second reason is that there is a need to extend farther the ADM1 model frame definition, in order to simulate processes under inhibition conditions (Chen et al., 2008). For example, Angelidaki et al. (1999) has proposed a Haldane-type substrate inhibition function in order to model the case of LCFA inhibition/toxic effects over the AD-population. More recently, Palatsi et al. (2010) proposed an inhibition-adsorption model to account for the LCFA inhibition/toxic effect, improving the standard ADM1 model fitting over slightly inhibited AD-populations (2-3 days of inhibition lag time).

Integrated modeling of activated sludge and AD wastewater has lead to the development of different modeling methodologies. Grau et al. (2007a) developed a new plant-wide modelling methodology for WWTPs, which does not require the development of specific transformers to interface the ADM and ASM models. On the other side, Zaher et al. (2007) and Nopens et al. (2009) have proposed interface units that try to preserve the element and charge balance. The integration of WWTP sub-units opens the possibility of global process optimization and to test control strategies and concurrent operation scenarios on a plant-wide scale.

1.4 The Uncertainty Challenge

Pieter Eykhoff (1974) defined a mathematical model as “a representation of the essential aspects of an existing system (or a system to be constructed) which presents knowledge of that system in usable form”. Since the AD process is quite complex, only essential aspects can be represented by explicit models. This approximation of reality implies that predictions derived from such a model can never be completely accurate. The EPA’s “Guidance on the Development, Evaluation, and Application of Environmental Models” (Gaber et al., 2009) proposes a taxonomy of uncertainties that may cause error in a model’s predictions.

Input uncertainty. There is typically uncertainty about data measurement, inconsistencies between measured values and those used by the model, and parameter value uncertainty.

Niche uncertainty. It may result when the model is used outside the domain for which it was originally developed (i. e. extrapolation problem) and/or it is build on several existing models with different spatial or temporal scales.

Framework uncertainty. An explicit model is a simplified abstraction of the reality. Even if there is no input and niche uncertainty, the predictions from the model will not equal those from the system because some factors that control the behavior of the system are unknown or were left outside of the model.

Apart from the above sources of uncertainty, complex systems may not always take the same value even if the same experimental conditions are repeated. Thus, when comparing the values of a deterministic model with the true values of the system, a residual variability may be present. Finally, uncertainty may arise from numerical approximations, called “code uncertainty”. An example is uncertainty analysis, where we wish to propagate uncertainty in inputs through the model in order to identify the most salient output uncertainties, regions of robustness, and important thresholds. One approach is to use a Monte Carlo sampling of inputs. The sampling error in the Monte Carlo estimates is an instance of code uncertainty. For a given model, one of the crucial elements is to determine how significant the single sources of uncertainty are over the results.

In practice, professionals deal with uncertainty implicitly by applying factors of safety provided by design codes, engineering textbooks or derived from personal experience. The consequence in plant operation practice is that conservative decisions are made, maintaining large safety margins in the plant designs and operation. An explicit treatment of uncertainty prevents information losses and stimulates rational decision making. When assumptions are clearly stated, possible flaws in the model used for decision making can promote new questions and a scientific habit of mind. The potential benefits of estimating or reducing the uncertainty in models used for a rational operation of biotechnological processes are many:

- Maintain the plant efficiency closer to its maximum by improved operation.
- Minimize the plant environmental impact.
- Increase the amount of organic waste that can be treated per unit process capacity.
- Decrease the frequency of gross process failures by increased process control.
- Run plants with less skilled personnel or decrease time devoted to plant management.
- The procedure for plant start-up can be shortened.
- Integrate the dynamics of the receiving organic wastes within the control of the plant.

The number of different approaches to model bio-systems is perhaps greater than the number of biological systems. In the present work, nonlinear models of ordinary differential equations are used. Those models are a combination of mechanistic and phenomenological models, frequently called “gray-box” models, characterized by parameters that may or may not be known with precision. The main question is how to represent explicitly precision or uncertanity in parameters. All the forms of uncertainty are uniquely represented and quantified by probability (O’Hagan and Oakley, 2004). In practice, however, experts may find it difficult to express their knowledge in probabilistic form. In particular, parameter value uncertainty elicitation is of great importance to achieving the above benefits relative to biotechnological process modeling.

Among many inferential choices, the Bayesian is particularly appealing for dealing with uncertainty. When a model parameter θ is represented by a random variable, expert's prior knowledge is expressed as a probability distribution, $p(\theta)$. This prior knowledge can be updated if information is introduced via the probability distribution for the data, $p(D|\theta)$, where D denotes the collected data. Since data is fixed, $p(D|\theta)$ is viewed as a function $L(\theta; D)$, known as the likelihood function. Bayes' theorem allows to combined data and prior information:

$$p(\theta|D) = \frac{p(D, \theta)}{p(D)} = \frac{p(D|\theta)\, p(\theta)}{p(D)}.$$

$p(\theta|D)$ is called the posterior distribution and reflects the probability of θ, given the observed evidence. The normalizing constant $p(D) = \int p(D, \theta) \mathrm{d}\theta$ is sometimes termed the marginal likelihood or just "evidence". Closed form solution of $p(\theta|D)$ are available only for few special cases. In practice, the posterior is numerically approximated by sampling from a probability distributions that is only proportional to the true posterior distribution.

Markov chain Monte Carlo (MCMC) sampling methods and the increasing computational power has made possible that Bayesian inference has become one of the leading theory frameworks in the field of physics and of biology (Hibbert and Armstrong, 2009). Some popular MCMC samplers are reviewed by Andrieu (2003): the Metropolis-Hastings sampler, the Gibbs sampler, the Slice sampler (a generalized form of the Gibbs sampler), the Hybrid Monte Carlo, the Adaptive MCMC sampler, the Particle Filters sampler, the Reversible jump MCMC sampler and the Simulated annealing sampler. Unfortunately, there has not been found any theoretical result to determine how many steps a Markov chain should have in order to reach its equilibrium (Murray, 2007). Anyway, there is an extensive literature on standard diagnostic tools (Adlouni et al., 2006) that could check the convergence properties of a chain (e. g., auto-correlation function, Gelman-Rubin statistics, etc.) and that could find analytical or coding errors in posterior simulators (Geweke, 2004). Bayesian inference is not immune to criticism (e.g. Gelman, 2008a; Kadane, 2008; Senn, 2008; Wasserma, 2008; Gelman, 2008b), since it gives an alternative definition of probability, which differs from the classical frequentist view.

1.5 Objectives and Questions

The underlying objective of the PhD work is to improve biotechnological processes, in particular the anaerobic co-digestion process, with the aid of modeling and uncertainty analysis. The objective is threefold:

1. Build a user-friendly simulation environment with direct access to powerful statistical methods and propose a new explicit model for the LCFA-inhibition process.

2. Identify gains and limitations when applying classical and Bayesian approaches for parameter uncertainty estimation.

3. Improve by simulation anaerobic co-digestion plant efficiency by explicitly accounting for input uncertainty.

In specific, the following questions shall be addressed:

- How to build a user-friendly simulation environment and how to use it? (Chapter 2)

- What explicit model is adequate to represent the LCFA-inhibitory process? How to quantify the overall cell-damage of sensible anaerobic populations? (Chapter 3)

- When to use classical frequentist and Bayesian inferential procedures? What is the quality of computed inferential results? (Chapter 3, 4)

- How to explicitly account for uncertainty, when optimizing the operation of an industrial co-digestion biogas plant? Under model uncertainty and multiplicity, what is the set of choices that are Pareto efficient in a multi-criteria analysis of the system? (Chapter 5) What is the payoff of an uncertanity reduction relative to a particular set of parameters? (Appendix A)

1.6 Thesis Outline

Chapter 2. Reviews some gains and limitations of popular simulation environments used for biotechnological process modeling and in particular for AD modeling. Our alternative simulation environment for continuous dynamic systems, called “virtual plant” (VP) toolkit is presented. Statistical tools used by the VP are described and a simple modeling case study of an industrial biogas plant is presented.

Chapter 3. Proposes two explicit models to answer a retrospective question regarding causes of LCFA-inhibition. ADM1 framework was used. Simple kinetics are considered to describe the bio-physics of the inhibitory process: i) adsorption of LCFA over granular biomass and ii) specific LCFA-degrader populations. In the first model, a commonly used non-competitive inhibition function is assumed. Contrary, in the second model, a new state variable is presented, which tries to account for a loss of cell-functionality induced by the adsorbed LCFAs. A comparison between the two models is performed and an explanatory analysis of the LCFA-inhibition process is given.

Chapter 4. A comparative study on how parameter uncertainty is estimated according to classical frequentist (linear) and Bayesian (non-linear) inferential procedures is performed. A modified wastewater treatment activated sludge model (ASM) was used for this purpose. Results were compared to evidence the strengths and weaknesses of both approaches.

Chapter 5. Presents a multi-criteria evaluation methodology for determining the operating strategies for biotechnological process models under an uncertainty and multiplicity. Based on the proposed uncertainty analysis, a reliability map is built for an industrial anaerobic co-digestion biogas plant for a given set of substrates mixture input loads.

Chapter 6. Briefly summarizes the major findings of the thesis by answering the above stated questions.

Appendix A. Presents a value of information analysis relative to the case study of Chapter 5. In particular, a partial expected value of perfect information analysis in computationally expensive models is tested; and an economic report on methane production is reported using this methodological framework.

Appendix B. Includes the source code of the VP and some screen-shots of the user-interface.

Bibliography

Adlouni, S. E., Favre, A.-C., Bobée, B., 2006. Comparison of methodologies to assess the convergence of Markov chain Monte Carlo methods. Computational Statistics & Data Analysis 50 (10), 2685 – 2701.

Andrieu, C., 2003. An introduction to MCMC for machine learning.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.13.7133

Angelidaki, I., Ellegaard, L., Ahring, B. K., 1999. A comprehensive model of anaerobic bioconversion of complex substrates to biogas. Biotechnology and Bioengineering 63 (3), 363–372.

Appels, L., Baeyens, J., Degrève, J., Dewil, R., 2008. Principles and potential of the anaerobic digestion of waste-activated sludge. Progress in Energy and Combustion Science 34 (6), 755 – 781.

Astals, S., Ariso, M., A., G., Mata-Alvarez, J., 2011. Co-digestion of pig manure and glycerine: Experimental and modelling study. Journal of Environmental Management 92 (4), 1091–1096.

Bastone, D. J., Keller, J., Angelidaki, I., Kalyuzhnyi, S. V., Pavlostathis, S. G., Rozzi, A., Sanders, W. T. M., Siegrist, H., Vavilin, V. A., 2002. Anaerobic digestion model no.1 (ADM1). Tech. rep., IWA Publishing , UK.

Boubaker, F., Ridha, B. C., 2008. Modelling of the mesophilic anaerobic co-digestion of olive mill wastewater with olive mill solid waste using anaerobic digestion model no. 1 (ADM1). Bioresource Technology 99 (14), 6565–6577.

Chen, Y., Cheng, J. J., Creamer, K. S., 2008. Inhibition of anaerobic digestion process: a review. Bioresource Technology 99 (10), 4044–4064.

Gaber, N., Foley, G., Pascual, P., Stiber, N., Sunderland, E., Cope, B., Nold, A., Saleem, Z., 2009. Guidance on the development, evaluation, and application of environmental models. Tech. rep., U.S. Environmental Protection Agency, EPA.

Galí, A., Benabdallah, T., Astals, S., Mata-Alvarez, J., 2009. Modified version of ADM1 model for agro-waste application. Bioresource Technology 100 (11), 2783–2790.

Gelman, A., 2008a. Objections to Bayesian statistics. Bayesian Analysis 3 (3), 445–450.

Gelman, A., 2008b. Rejoinder. Journal of the American Statistical Association 103 (482), 449–451.

Geweke, J., 2004. Getting it right: Joint distribution tests of posterior simulators. Journal of the American Statistical Association 99 (467), 799–804.

Grau, P., de Gracia, M., Vanrolleghem, P. A., Ayesa, E., 2007a. A new plant-wide modelling methodology for WWTPs. Water Research 41 (19), 4357–4372.

Gujer, W., 2002. Microscopic versus macroscopic biomass models in activated sludge systems. Water Science and Technology 45 (6), 1–11.

Hejnfelt, A., Angelidaki, I., 2009. Anaerobic digestion of slaughterhouse by-products. Biomass and Bioenergy 33 (8), 1046–1054.

Henze, M., Gujer, W., Mino, T., van Loosdrecht, M., 2000. Activated sludge models ASM1, ASM2, ASM2d and ASM3. Tech. rep., IWA Publishing. London, UK.

Hibbert, D., Armstrong, N., 2009. An introduction to Bayesian methods for analyzing chemistry data: Part ii: A review of applications of bayesian methods in chemistry. Chemometrics and Intelligent Laboratory Systems 97 (2), 211 – 220.

Kadane, J. B., 2008. Comment on article by Gelman. Bayesian Analysis 3, 455–458.

Lübken, M., Wichern, M., Schlattmann, M., Gronauer, A., Horn, H., 2007. Modelling the energy balance of an anaerobic digester fed with cattle manure and renewable energy crops. Water Research 41 (18), 4085–4096.

López, I., Borzacconi, L., 2010. Modelling of slaughterhouse solid waste anaerobic digestion: Determination of parameters and continuous reactor simulation. Waste Management 30 (10), 1813–1821.

Murray, I., 2007. Advances in markov chain monte carlo methods. Ph.D. thesis, University College London.

Nopens, I., Batstone, D. J., Copp, J. B., Jeppsson, U., Volcke, E., Alex, J., Vanrolleghem, P. A., 2009. An ASM/ADM model interface for dynamic plant-wide simulation. Water Research 43 (7), 1913–1923.

O'Hagan, A., Oakley, J. E., 2004. Probability is perfect, but we can't elicit it perfectly. Reliability Engineering & System Safety 85 (1-3), 239–248.

Palatsi, J., Illa, J., Prenafeta-Boldú, F. X., Laureni, M., Fernandez, B., Angelidaki, K., Flotats, X., 2010. Long-chain fatty acids inhibition and adaptation process in anaerobic thermophilic digestion: batch tests, microbial community structure and mathematical modelling. Bioresource Technology 101 (7), 2243–2251.

Picioreanu, C., Batstone, D. J., van Loosdrecht, M. C. M., 2005. Multidimensional modelling of anaerobic granules. Water Science and Technology 52 (1-2), 501–507.

Senn, S., 2008. Comment on article by Gelman. Bayesian Analysis 3, 459–462.

Wasserma, L., 2008. Comment on article by Gelman. Bayesian Analysis 3 (3), 459–462.

Woodgate, S., van der Veen, J., 2004. The role of fat processing and rendering in the European Union animal production industry. Biotechnology, Agronomy, Society and Environment 8 (4), 283–294.

Zaher, U., Grau, P., Benedetti, L., Ayesa, E., Vanrolleghem, P., 2007. Transformers for interfacing anaerobic digestion models to pre- and post-treatment processes in a plant-wide modelling context. Environmental Modelling & Software 22 (1), 40 – 58.

Zaher, U., Li, R., Jeppsson, U., Steyer, J.-P., Chen, S., 2009. GISCOD: General integrated solid waste co-digestion model. Water Research 43 (10), 2717–2727.

2 Methods. Development of the Virtual Plant Toolkit

2.1 Abstract

The virtual plant (VP) toolkit, developed during the present work, is a simulation environment for continuous dynamic systems, which are described in terms of ordinary differential/algebraic equations (ODEs/DAEs). The system-equations are expressed in an Excel-sheet, which symbolic representation is converted in C-mex language by auto-code generation. The Simulink block-oriented environment of MATLAB is used for simulation. Parameter inference and process-optimization routines are integrated in the VP-toolkit. Moreover, auto-code is generated for two external applications: Graphviz and GEM-SA. The first visualizes the reaction pathways as a directed graph and the last performs global sensitivity analysis. Some of the VP-functionalities are shown on a simple case study modeling of an anaerobic digestion biogas plant.

List of Abbreviations

VP	Virtual Plant
ODEs	Ordinary Differential Equations
DAEs	Differential Algebraic Equations
SUNDIALS	SUite of Nonlinear and DIfferential/ALgebraic equation Solvers
FD	Finite Difference
MCMC	Markov Chain Monte Carlo
MLE	Maximum Likelihood Estimation
MAP	Maximum A posteriori Probability
SS	Scatter-Search
GEM-SA	Gaussian Emulator Machine for Sensitivity Analysis
GSA	Global Sensitivity Analysis
SA	Sensitivity Analysis
DRAM	Delayed Rejection Adaptive MCMC
FIM	Fisher Information Matrix
MH	Metropolis-Hastings
AM	Adaptive MCMC
DR	Delayed Rejection
GP	Gaussian Process
LCFA	Long Chain Fatty Acids
AHM	Anaerobic Haldane Model
AMM	Anaerobic Monod Model

PNM	Pathway Network Model
GMM	Gaussian Mixture Model

2.2 Introduction

Ordinary differential/algebraic equations (ODEs/DAEs) are the building-blocks for many mathematical models in sciences and engineering. When describing wastewater treatment processes with ODEs, the model is integrated beginning with specified initial values and, typically, events are scheduled dynamically as the simulation proceeds. The process of system-simulation can be divided in two steps: mathematical system-description and its numerical integration. A simulation environment helps to implement those two steps, facilitating the re-use of knowledge in process models, while maintaining a high performance in simulation speed and accuracy. However, it is not straightforward to build a simulation environment where code-maintenance and simulation-performance are both maximized. For example, if the mathematical system-description is coded in a programming language like Fortran or C, the result is a very-fast code execution, but low re-usability since only skilled programmers have enough knowledge to modify or maintain the code.

During the last decade, a large number of simulation environments have been built with the aim of expanding the practice of model-simulation to "occasional" programmers. The long summary list of software for simulation of the "Systems Biology Markup Language" is an example. In particular, in the field of wastewater treatment, commercial simulators like GPS-X[1], Biowin[2], Simba [3] and WEST[4], among others, were built to satisfy the criteria of re-usability and simulation performance. An example of an open-source counterpart to those commercial tools is the wastewater library of OpenModelica (Reichl, 2003). All those simulation packages use some sort of graphical block diagramming (e. g., Simulink blocks), which makes it easy to visualize the units (i. e. sub-models) of a complex system.

Process-simulation is the building-block for further analysis and optimization of the process. Thus, it is important that a simulation environment has (or is linked to) a powerful statistical or optimization set of methods. The majority of ad-doc simulation environments miss the analysis flexibility provided by libraries of algorithms and mathematical tools such as SciPy, Octave/MATLAB, R, and others. Simba is an exception

[1] © 2012 Hydromantis Environmental Software Solutions
[2] © 2003 EnviroSim
[3] © 2012 ifak system GmbH
[4] © 2012 MOSTforWater

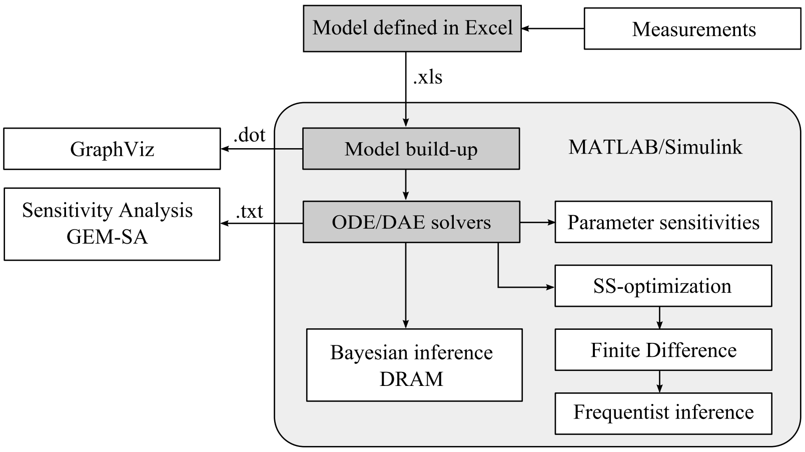


Figure 2.2.1: Virtual Plant toolkit structure.

since it is implemented within the MATLAB/Simulink environment, but its commercial license makes its use restricted. A very interesting option is the simulation environment ACADO (Automatic Control and Dynamic Optimization) toolkit for MATLAB (Houska et al., 2011), but it misses the block-diagramming representation.

In order to alleviate some of the above limitations, we have built our own modeling/simulation environment: the Virtual Plant (VP) toolkit. The main structure of the VP is shown in Figure 2.2.1. Setting of model-parameters, events, process-data and mathematical process-description are stored in Excel-sheets. In this way, a low-trained user can test or modify models and use available data to learn about the process at hand. The Excel-sheets are loaded in MATLAB and translated in a C-mex code, which is compiled and used in a Simulink-block. Numerical performance is guaranteed by the ODE-suite of MATLAB (i. e. ode15s, ode45, etc.). Another option available is the "suite of nonlinear and differential/algebraic equation solvers" (SUNDIALS), but simulation performance is reduced because the code is not compiled. The SUNDIALS allowes for the solution of differential algebraic equations (DAEs) too. Because model-simulation is performed in MATLAB, a wide variety of statistical tools are available.

When measurements of the process are available, a common task is model calibration. There are two general methods for parameter inference: frequentist and Bayesian. Both are implemented in the VP toolkit (see Figure 2.2.1). The common denominator of those two procedures is the maximum likelihood estimation (MLE) of the unknown parameter set (Chapter 4), which can be found numerically using optimization methods. The VP uses a scatter-search (SS) global optimization algorithm (Larrosa, 2008; Rodriguez-Fernandez et al., 2006a) discussed in Sub-section 2.3.1. MLE is not only used for summarizing observed data, but it is also useful for testing hypotheses or constructing frequentist (linear) confidence regions. The finite difference (FD) method called "Adaptive Robust Numerical Differentiation" (D'Errico, 2006) was included in the VP since frequentist confidence regions are based on a Hessian matrix approximation at the MLE-value. On the other hand, Bayesian inferential procedure needs an efficient Markov Chain Monte Carlo (MCMC) sampler. In the VP-toolkit, the "Delayed Rejection Adaptive MCMC" (DRAM) sampler (Laine, 2008) was included.

It is often valuable to know how the solution changes with model-parameters. VP implements a local and a global method for parameter sensitivities evaluation. The local method is based on sensitivity function trajectories (Dochain and Vanrolleghem, 2001; Petersen et al., 2001; Marsili-Libelli et al., 2003), which are evaluated exactly if the Symbolic toolbox of MATLAB is available or approximated numerically by the CVODES solver from the SUNDIALS-suite. The global sensitivity analysis (GSA) is based on an external free-application called "Gaussian Emulator Machine for Sensitivity Analysis (GEM-SA)" (Oakley and O'Hagan, 2004). VP generates automatically the necessary input files for the GEM-SA application. Another VP-external application is Graphviz, a visualization tool that can represent structural information as diagrams of abstract graphs and networks. The Graphviz-code is automatically generated by the VP and reaction pathways can be displayed or exported by Graphviz.

In this chapter, we present in detail the SS-optimization routine (Sub-section 2.3.1), the MCMC-sampler (Sub-section 2.3.3) and the probabilistic GSA method (Sub-section 2.3.4), while we postpone the presentation of frequentist and Bayesian inferential procedures to Chapter 4. A short example of the VP-toolkit is given in Section 2.4, where we model an industrial anaerobic digestion reactor.

2.3 Methods

In the following sub-sections, the different routines/tools/procedures used for the VP built-up are described.

2.3.1 Scatter-Search Global Optimization

Scatter-search (SS) optimization is a class of evolutionary algorithm based on generalized path constructions in Euclidean space and strategic decision rules. The SS algorithm were developed by the "Process Engineering Group" of the "Instituto de Investigaciones Marinas", Spain (Larrosa, 2008; Rodriguez-Fernandez et al., 2006a) and implemented in

MATLAB as a free toolbox. The SS-optimization is based on the following six rules, or better methods (Larrosa, 2008):

Method 1. Diversification generation method generates proposal vectors for the initial set P. This is accomplished by dividing the suggested parameter space into sub-ranges, which are kept fixed during the optimization. Anytime DGM is called, it randomly selects a sub-range based on its past "degree" of exploration. When the range scale is wide, the logarithmic distribution is better suited for fast convergence than the uniform one. The proposal vector is randomly chosen from the selected sub-range.

Method 2. Reference set build method builds the reference set S. It has two strategies: the first one combines proposals based on quality and diversity when building S (fast convergence), whereas the second focuses only on diversity (less simulations). The first strategy selects two sub-sets of vectors from P: the best $b/2$ solutions from P (quality) and the $b/2$ selections that maximize the minimum distance between the remaining candidate solution of P and the solutions currently in S (diversity). The second strategy, instead of selecting the $b/2$ best solutions, only set in S the lower, the middle and the upper bound vector. S is completed by b vectors using the diversity strategy.

Method 3. Subset generation and combination method operate on S, to produce several subsets of its solutions as a basis for creating combined solutions. Generation involves selecting all pairs of proposals in S. Repeated pairs are avoided by using a memory record of the already used combinations. Combination is based on hyper-rectangles generated from the pairs, called "parent". During combination every generated vector, called *child*, is compared with its parent. If the child outperforms its parent in terms of quality a new hyper-rectangle defined by the distance between the parent and the child is constructed. The procedure is repeated until there is a gain in performance, with the only difference that the hyper-rectangle is increased by two times, since it is a very promising area of search.

Method 4. Reference set update method updates S by working on the high-quality and diverse proposals and the new combined solutions from *Method 3*. There are two types of filters: a *distance filter* that prevents similar proposals to enter S through the use of a threshold value and a *diversity filter* that prevents vectors in the same flat area to join S. Both the filters restrict the incorporation of proposals that contribute only slightly to the quality and diversity of the current S. *Intensification* is a procedure that stores in a secondary reference set $\hat{S}$ the proposals that were rejected by the distance and diversity filters. Every i_{freq} number of iterations, the proposals from $\hat{S}$ are combined trough *Method 3* with the proposals from S in order to speed-up the convergence to the global optimum.

Method 5. Improvement method performs a local search (e. g., sequential quadratic programming method) starting from a carefully selected solution. A *merit filter* avoids *Method 5* to be applied to a low-quality solution and a distance filter prevents *Method 5* application to a solution close to other for which the *Method 5* was applied in previous iterations. Another constrain in order to avoid performing many local searches from similar initial proposals is to fix a minimum number of function evaluations between two local searches. After this minimum number of evaluations, a local search is performed if the algorithm finds a better solution.

Method 6. Rebuilding method is applied whenever *Method 3* fails to provide any new proposal to S. In the first step, g worst solutions from S are deleted, the best solution is selected as the center of gravity and connected trough segments to the remaining $b-g-1$ solutions of S. In the second step, m new proposals are generated by the *Step 1* method, which are connected once again to the center of gravity. Finally, the new proposals that are included in S are those which maximize the orthogonality with the solutions already in S.

It is interesting to note that the IM method is needed only if high quality outcomes are desired, but can be omitted because of the problem's nature or because of the high computation costs associated with the function evaluations.

For our applications, the SS-optimization is applied to estimate the maximum a posteriori probability (MAP) estimate, i. e. the mode of the posterior distribution $p(\theta|D)$. In particular, when the prior is uninformative the posterior is entirely defined by the likelihood function $L(\theta;D)$ and thus the MAP estimate is called maximum likelihood estimation (MLE).

2.3.2 Frequentist inference

Briefly, Frequentist estimation of parameter value uncertainty, relay on the estimation of the covariance matrix $\mathbf{C}$. Suppose that data $D = d_1, ..., d_n$ is available. If the additive noise model

$$d_i = f_i\left(\theta\right) + \epsilon_i, \quad \epsilon \sim Norm\left(0,\, \sigma^2 \mathbf{I}_n\right),$$

is selected, where $f(\theta)$ is a linear model of process dynamics and ϵ is an independent Gaussian error term with a homoscedastic variance error parameter, σ^2, then the $100\,(1-\alpha)\,\%$ confidence region for $\theta \in \mathbb{R}^p$ is

$$\left\{\theta : \left(\theta - \hat{\theta}\right)^T \mathbf{C}^{-1}\left(\theta - \hat{\theta}\right) \leq pF^{\alpha}_{p,n-p}\right\}, \tag{2.3.1}$$

where $F^{\alpha}_{p,n-p}$ is the Fisher-Snedecor distribution. The covariance matrix $\mathbf{C}$ is estimated as $\mathbf{C}_J = s^2\left(\mathbf{J}^T\mathbf{J}\right)^{-1}$, where $\hat{\mathbf{J}} = [\partial f/\partial\theta]_{\hat{\theta}}$ is the Jacobian matrix of the model estimated at the MLE-parameter nominal location $\hat{\theta}$ and $s^2 = SS\left(\hat{\theta}\right)/(n-p)$ is an unbiased approximation of the residual variance σ^2. SS is the sum of squares function

$$SS\left(\hat{\theta}\right) = \sum_{i=1}^{n}\left(d_i - f_i\left(\hat{\theta}\right)\right)^2.$$

Note that $\mathbf{C}_J\left(\hat{\theta}\right) = \mathbf{FIM}^{-1}$ is the inverse of the Fisher Information Matrix (FIM). It is also possible to estimate $\mathbf{C}$ trough the Hessian matrix $\mathbf{H} = \left[\partial^2 SS(\theta)/\partial\theta\partial\theta^T\right]_{\hat{\theta}}$, since $\mathbf{C}_H = 2s^2\mathbf{H}^{-1}$. Theoretically, the estimation of $\mathbf{C}$ trough $\mathbf{C}_J$ or $\mathbf{C}_H$ will differ only if for some numerical or other reason the $SS(\theta)$ minimum is not reached. This is because $\mathbf{H}$ and $\mathbf{FIM}$ differ by a term involving the curvature of the $SS(\cdot)$ function. Finally, the individual parameter confidence interval is estimated as $\delta_i = \pm t^{1-(\alpha/2)}_{n-p}\sqrt{C_{ii}}$, where $t^{1-(\alpha/2)}_{n-p}$ is the two-tails Student's t distribution (Seber and Wild, 1989).

When $f(\theta)$ is nonlinear in parameters, Eq. (2.3.1) gives only a linear approximation of the uncertainty in the vicinity of the MLE solution. The Frequentist approximation may be a reasonable approach for very informative data sets (i. e. large enough n), but for sparse data and non-Gaussian measurement error models the confidence region given in Eq. (2.3.1) may underestimate parameter-uncertainty (Vrugt and Bouten, 2002). Moreover, when data are sparse, the condition number should be checked in order to avoid high approximation errors during the operation of inversion.

Correct estimation of the covariance matrix $\mathbf{C}$ implies a robust numerical procedure for the approximation of the Jacobian or Hessian matrix. Apart of symbolic explicit solutions, finite difference (FD) approximations are generally used. The main problem for FD methods that there is no general solution for nonlinear functions on how to evaluate the perturbation parameter $\mathbf{h} \approx \partial\theta$ (Press et al., 1997). Model simulation and parameter estimation software packages like PEAS (Checchi et al., 2007) or SIMAQUA (Reichert et al., 1995) use an arbitrary fixed perturbation parameter value, since they assume that it would be acceptable for most applications. Pauw (2005) has compared different estimation techniques for the Hessian matrix and has shown empirically that the influence of $\mathbf{h}$ is parameter specific. Thus, fixing a global perturbation parameter value is not a very good choice.

Dochain and Vanrolleghem (2001) has given in their book a practical advice: to compare the results for halving the perturbation value $\mathbf{h}$ until the results are sufficiently close. For example, this advice is followed by Checchi and Marsili-Libelli (2005). In our VP, the free-MATLAB toolbox called "Adaptive Robust Numerical Differentiation" (D'Errico, 2006) was used to estimate $\mathbf{H}$. The estimation routine is based on a FD, fourth-order Romberg-extrapolation method with an adaptive routine for the determination of the step-size-perturbation parameters.

2.3.3 Delayed Rejection Adaptive MCMC

The adaptive version of the random walk Metropolis-Hastings (MH) algorithm with a Gaussian proposal distribution called "Delayed Rejection Adaptive MCMC" (DRAM) has been provided as open-source MATLAB toolbox by Marko Laine Laine (2008); Laine and Tamminen (2008). In order to understand how DRAM algorithm works, a short description of the MH algorithm is presented first.

The MH algorithm draws samples from a distribution that is only known up to a constant. Random numbers are generated from a distribution with a target distribution $\pi(\theta)$ that is equal to or proportional to a proposal function p. In most cases the target will be the posterior distribution for the model unknowns, $\pi(\theta) = p(\theta \mid D)$. The algorithm generates a discrete random process called the Markov chain, in which each state θ_{t+1} depends only on the previous state θ_t. To generate the chain the algorithm proceeds as follows:

Step 1. Assume an initial value θ_t and set a proposal (or prior) distribution $q_1(\theta)$ of the target $\pi(\theta)$.

Step 2. Draw a new sample, θ^*, from a proposal distribution $q_1(\theta)$.

Step 3. Calculate the first stage acceptance probability $\alpha_1(\theta_t \mid \theta^*) = \min\{1, ab\}$, where the likelihood ratio is $a = \pi(\theta^*)/\pi(\theta_t)$ and the ratio of the proposal density is $b = q_1(\theta^* \mid \theta_t)/q_1(\theta_t \mid \theta^*)$[5].

Step 4. If $ab \geq 1$ than $\theta_{t+1} = \theta^*$, else $\theta_{t+1} = \begin{cases} \theta^* & \text{with probability } \alpha \\ \theta_t & \text{with probability } 1-\alpha \end{cases}$ (drawing from a uniform distribution).

[5] $b = 1$ because q_1 is a Gaussian proposal symmetric-type of probability distribution.

Step 5. Steps 2, 3 and 4 are repeated until a desired number of samples is reached.

The Markov chain has to be run until the initial state is "forgotten". These samples, which are discarded, are known as "burn-in*"*. The remaining set of accepted values represent a sample from the distribution $\pi(\theta)$. The limitation of the MH algorithm is that in order to converge rapidly to the target distribution, it is required that $\pi(\theta^*) \approx q_1(\theta_t \mid \theta^*)$. Since in practice the proposal is often unknown, the variance parameter σ^2 of a Gaussian proposal is used to optimize the "burn-in" period by α_1, which is the fraction of proposed samples that is accepted in a window of the last samples.

There are two cases when the chain will converge very slowly to $\pi(\theta)$ (poor mixing). First, if σ^2 is too small, α_1 will be high but successive samples will move around the space slowly. On the other hand, if σ^2 is too large,α_1 will be very low because the proposals are likely to land in regions of much lower probability density (very low a).

The DRAM algorithm, enhances chain mixing using an adaptive MC (AM) algorithm, which is applied on the covariance matrix of the proposal. In the following, the AM algorithm is presented as given by Laine (2008):

Step 1. Start from an initial value θ_0 and give an initial proposal covariance $C = C_0$, that could be find by an initial fit of $SS(\theta)$ and the relative Jacobian matrix estimation $C_0 = \left(J^T J\right)^{-1} s^2$. This sampler initialization is valid, since these samples will be discarded in the "burn-in". Select a covariance scaling factor s, a small number ϵ for regularizing the covariance (prevent matrix singularity), and an initial non-adapting period n_0.

Step 2. At each step, propose a new θ^* from a Gaussian distribution centered at the current value $N(\theta_t, C)$.

Step 3. Accept or reject according to the MH acceptance probability α_1.

Step 4. After an initial period of simulation, say for $t \geq n_0$, adapt the proposal covariance matrix using the chain generated so far by $C = \mathrm{cov}(\theta_0, \theta_1, \ldots, \theta_t)\, s + I\varepsilon$. Adapt from the beginning of the chain or with an increasing sequence of values. Adaptation can be done at fixed or random intervals.

Step 5. Iterate from Step 2 until enough values have been generated in order to approximate $\pi(\theta)$.

As can be noted, AM depends on the past values generated in the chain, thus the problem is to find a feasible initial proposal in order to start the adaptation process. One way to solve this proposal initialization problem, is to apply a "delayed rejection method"

(DR). This method exploits the rejected proposals θ^* from the MH-step, in order to make a second proposal θ^{**}: in this way, proposal rejection is delayed. In DR a second stage acceptance probability is defined as

$$\alpha_2\left(\theta_t,\, \theta^* \mid \theta^{**}\right) = \min\left\{1,\ \frac{\pi\left(\theta^{**}\right) q_1\left(\theta^{**} \mid \theta^*\right) q_2\left(\theta^{**},\, \theta^* \mid \theta_t\right)\left[1 - \alpha_1\left(\theta^{**} \mid \theta^*\right)\right]}{\pi\left(\theta_t\right) q_1\left(\theta_t \mid \theta^*\right) q_2\left(\theta_t,\, \theta^* \mid \theta^{**}\right)\left[1 - \alpha_1\left(\theta_t \mid \theta^*\right)\right]}\right\},$$

where θ^{**} is drawn from a given $q_2\left(\theta_t,\, \theta^* \mid \cdot\right)$ and the acceptance test is performed in the same way as in MH. The procedure can be iterated further for higher-stage proposals, even if for most cases one or two tries are enough. In this way, rejected values drives a local adaptation to the current location of the target distribution $\pi\left(\theta\right)$. Another strength of this local adaptation, is that positivity constraints in the unknowns of the model can be implemented, since DR would reject proposed values which are non-positive.

Note that the proposal ratio of q_1 for the second stage acceptance, for example, do not cancel out and it has to be calculated explicitly. If Gaussian independent proposal is assumed, then the proposal ratio of q_1 for α_2 is simply stated as

$$\frac{q_1\left(\theta^{**} \mid \theta^*\right)}{q_1\left(\theta_t \mid \theta^*\right)} = \exp\left\{-\frac{1}{2}\left(\theta^{**} - \theta^*\right)^T \left(C_t\right)^{-1}\left(\theta^{**} - \theta^*\right) + \frac{1}{2}\left(\theta_t - \theta^*\right)^T \left(C_t\right)^{-1}\left(\theta_t - \theta^*\right)\right\}$$

and the posterior ratio need in MH is

$$\frac{\pi\left(\theta^*\right)}{\pi\left(\theta_t\right)} = \exp\left\{-\frac{1}{2\sigma^2}\left(SS\left(\theta^*\right) - SS\left(\theta_t\right)\right) + \frac{1}{2}\left(SS_{pri}\left(\theta^*\right) - SS_{pri}\left(\theta_t\right)\right)\right\},$$

where the error variance σ^2 is assumed homoscedastic.

2.3.4 Probabilistic Sensitivity Analysis with GEM-SA

Saltelli et al. (2006) in their review of sensitivity analysis (SA) tools, concluded, that in spite of considerable development in this filed during the last forty years, only primitive SA tools are used in practice, manly based on the local derivatives or ‘‘one-factor-at-a-time’’ approaches. They demonstrate that in the context of model corroboration, methods based on local sensitivity analysis are illicit and unjustified, unless the model under analysis is proved to be linear Cariboni et al. (2007); Saltelli et al. (2006). Finally, they show that variance-based measures for glabal SA enter in what they call a “good practice methods”, making the factors importance ranking (e. g., sensitivity ranking of parameters) univocal. As it is well known, the main disadvantage of the local methods is to not account for interactions between variables and the local sensitivity indexes are related to a fixed nominal point in the space of parameters.

A global SA tool used to work in tandem with the VP: the "Gaussian Emulation Machine for Sensitivity Analysis" (GEM-SA) free-software (Kennedy and O'Hagan, 2001; Oakley and O'Hagan, 2004). This Bayesian approach to probabilistic SA is both robust and highly efficient, allowing sensitivity analysis to be applied to expensive models. The method is appropriate for a class of models that can respond continuously to changes in its inputs. It implements statistical analysis of uncertainty in the outputs of computer models, using Gaussian process (GP) emulation Kennedy (2004). An emulator duplicates the functions of one system $y = f(\mathbf{x})$ using a different system $\hat{y} = f_{GP}(\mathbf{x})$, so that the second system behaves like the first one. The advantage of building an emulator $f_{GP}(\cdot)$ is to perform much cheaper evaluates than the original model $f(\cdot)$, while accounting for the uncertainty in the approximation. Moreover, the number of training inputs required to build a GP based-emulator is very small, and thus it needs only few computer model evaluations of the original code.

In GEM-SA, the following GP prior probability distribution is adopted:

$$[f_{GP}(\cdot) \mid \beta,\, \sigma,\, r] \sim Norm\left(m(\cdot)\,,\, \sigma^2 c(\cdot,\cdot)\right),$$

where the mean and the correlation functions take the form

$$\begin{aligned} m(\cdot) &= \beta^T \mathbf{h}(\cdot)\,, \\ c(\cdot,\cdot) &= \exp\left\{-\left(\mathbf{x}-\grave{\mathbf{x}}\right)^T \mathbf{R}\left(\mathbf{x}-\grave{\mathbf{x}}\right)\right\} \qquad \forall\, \mathbf{x},\, \grave{\mathbf{x}} \in \mathbf{X}. \end{aligned}$$

Here $\mathbf{h}(\mathbf{x})$ is a vector of q known regression functions, β is a vector of regression coefficients, σ^2 is the variance and $\mathbf{R} = \operatorname{diag}(r_i)$ is a diagonal matrix of (positive) roughness parameters. Typical regression functions are $\mathbf{h}(\mathbf{x}) = 1$ or $\mathbf{h}(\mathbf{x})^T = \left[1,\, \mathbf{x}^T\right]$.

After the prior specification of the GP emulator is given, carefully selected design points $\mathbf{x}_1, ..., \mathbf{x}_n$ evaluated by $f(\cdot)$ are used to obtain the corresponding outputs $\mathbf{y} = y_1, \ldots, y_n$. Note that the discussed GP is a multiple-input, single-output system. Given these data, the marginal posterior process for the model $f(\cdot)$, conditional upon the roughness matrix $\mathbf{R}$ is

$$[f(\cdot) \mid \mathbf{R},\, \mathbf{y}] \sim t_{n-q}\left(m^*(\mathbf{x})\,,\, \hat{\sigma}^2 c^*(\cdot,\cdot)\right), \tag{2.3.2}$$

which is a Student's distribution, where

$$
\begin{aligned}
m^*(\mathbf{x}) &= h(\mathbf{x})^T\hat{\beta} + \mathbf{t}(\mathbf{x})^T\mathbf{A}^{-1}\left(y - \mathbf{H}\hat{\beta}\right),\\
c^*(\mathbf{x},\,\dot{\mathbf{x}}) &= c(\mathbf{x},\,\dot{\mathbf{x}}) - \mathbf{t}(\mathbf{x})^T\mathbf{A}^{-1}\mathbf{t}(\mathbf{x}) +\\
&\quad + \left(\mathbf{h}(\mathbf{x})^T\mathbf{t}(\mathbf{x})^T\mathbf{A}^{-1}\mathbf{H}\right)\left(\mathbf{H}^T\mathbf{A}^{-1}\mathbf{H}\right)^{-1}\left(\mathbf{h}(\dot{\mathbf{x}})^T - \mathbf{t}(\dot{\mathbf{x}})^T\mathbf{A}^{-1}\mathbf{H}\right)^T.
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{t}(\mathbf{x})^T &= \left[c(\mathbf{x},\,\mathbf{x}_1),\ldots,\,c(\mathbf{x},\,\mathbf{x}_n)\right],\\
\mathbf{H}^T &= \left[\mathbf{h}(\mathbf{x}_1)^T,\ldots,\,\mathbf{h}(\mathbf{x}_n)^T\right],\\
\mathbf{A} &= \begin{bmatrix} 1 & c(\mathbf{x}_1,\,\mathbf{x}_2) & \cdots & c(\mathbf{x}_1,\,\mathbf{x}_n)\\ c(\mathbf{x}_2,\,\mathbf{x}_1) & 1 & & \vdots\\ \vdots & & \ddots & \\ c(\mathbf{x}_n,\,\mathbf{x}_1) & \cdots & & 1 \end{bmatrix},\\
\hat{\beta} &= \left(\mathbf{H}^T\mathbf{A}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{A}^{-1}\mathbf{y},\\
\hat{\sigma}^2 &= \mathbf{y}^T\left\{\mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{H}\left(\mathbf{H}^T\mathbf{A}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{A}^{-1}\right\}\mathbf{y}\frac{1}{n-q-2}.
\end{aligned}
$$

The hyper-parameters $\hat{\beta}$ and $\hat{\sigma}^2$ are generalized least squares estimators of β and σ^2, so that the only hyper-parameters to guess are the roughness coefficients $r_i \in [0,\,1]$. A high value of the roughness hyper-parameter indicates that the emulated function responds strongly to quite small change in inputs and this implies that many more data are needed to accurately emulate a rough function. In practice, the roughness coefficients r_i are estimated from a set of training data, which are subsequently validated by predicting new data points.

Once the emulator is build, it can be used to perform probabilistic SA of the original model (Oakley and O'Hagan, 2004). In order to understand how probabilistic SA works, some important measures of variance-based SA has to be introduced.

The uncertainty of the input $\mathbf{X}$ is expressed by some probability distribution g, such that the elements of $\mathbf{X}$ are mutually independent. The sample vector is denoted as $\mathbf{x} = \{x_1,\ldots,\,x_d\}$ and the sub-vector $(x_i,\,x_j,\ldots,\,x_p)$ is denoted as $\mathbf{x}_p$ where p is a set of indexes. The sub-vector of $\mathbf{x}$ containing all elements except x_i is defined as $\mathbf{x}_{\sim i}$.

As a starting point to derive useful SA measures, consider the following function decomposition

$$y = f(\mathbf{x}) = E(Y) + \sum_{i=1}^{d} z_i(x_i) + \sum_{i<j} z_{i,j}(\mathbf{x}_{i,j}) + \ldots + z_{1,2,\ldots,d}(\mathbf{x}), \tag{2.3.3}$$

where

$$\begin{aligned} z_i(x_i) &= E(Y \mid x_i) - E(Y) \\ z_{i,j}(\mathbf{x}_{i,j}) &= E(Y \mid \mathbf{x}_{i,j}) - z_i(x_i) - z_j(x_j) - E(Y) \end{aligned}$$

and so on. The function is decomposed in a sum of main effects $z_i(x_i)$ and first-order joint effects $z_{i,j}(\mathbf{x}_{i,j})$ (even called "interactions") and so on components. Variance-based methods of SA quantify the sensitivity of the output Y to the model input in terms of a reduction in the variance of Y. Thus, measuring the expected amount $V_i = \text{var}\{E(Y \mid X_i)\}$ by which the uncertainty in Y will be reduced if we learn the true value of x_i, quantify the sensitivity of the output Y to the model input. To normalize the scale of the V_i measure, we simply divide by $\text{var}(Y)$ and thus

$$S_i = \frac{V_i}{\text{var}(Y)} = \frac{\text{var}(z_i)}{\text{var}(Y)}, \tag{2.3.4}$$

referred to as the main effect index of x_i. In a similar way, the first-order joint effect index $S_{i,j}$ can be defined, as

$$S_{i,j} = \frac{\text{var}\{E(Y \mid \mathbf{X}_{i,j})\}}{\text{var}(Y)} = \frac{\text{var}\{z_i + z_j + z_{i,j}\}}{\text{var}(Y)}. \tag{2.3.5}$$

In general, $V_p = \text{var}\{E(Y \mid \mathbf{X}_p)\}$ is the expected reduction in variance that is achieved when we learn $\mathbf{x}_p$. Another, useful measure is the total effect index of x_i, $S_{Ti} = 1 - S_{-i}$, which represents the sum of all main, interaction and higher order terms in which an input is involved. It is interesting to note, that the propriety of mutual independence of $\mathbf{X}$ permits to decompose the variance of Y into terms relating to the main effects and various interactions between the input variables

$$\text{var}(Y) = \sum_{i=1}^{d} W_i + \sum_{i<j} W_{i,j} + \ldots + W_{1,\ldots,d}, \tag{2.3.6}$$

where $W_p = \text{var}\{z_p\}$. Thus, we have that $W_i = V_i$ and that $V_{i,j} = W_i + W_j + W_{i,j}$, where could be found that $\sum_{i=1}^{d} S_i \leqslant 1 \leqslant \sum_{i=1}^{d} S_{Ti}$, with equality only if all the interactions are zero. In this way, the difference between S_i and S_{Ti} is an indicator of the presence or absence of joint effects or higher order interactions for a given input x_i.

The inference of the main effects, joint effects and variance through the use of a GP emulator is straight forward, since for a normal or an uniform input distribution g, constant or linear $\mathbf{h}(\mathbf{x})$ and Gaussian covariance $c(\cdot,\cdot)$, it is possible to evaluate these SA measures analytically. Consider the case of inference of

$$E(Y \mid \mathbf{x}_p) = \int\limits_{X_{-p}} f_{GP}(\mathbf{x})\, \mathrm{d}g_{-p|p}(\mathbf{x}_{-p} \mid \mathbf{x}_p)\,.$$

Since this is a linear functional of $f_{GP}(\mathbf{x})$, its posterior will be a Student's distribution, which posterior mean can be derived as

$$E^*\{E(Y \mid \mathbf{x}_p)\} = R_p(\mathbf{x}_p)\,\hat{\beta} + T_p(\mathbf{x}_p)\,\mathbf{e},$$

where

$$\begin{aligned} R_p &= \int\limits_{X_{-p}} \mathbf{h}(\mathbf{x})^T \,\mathrm{d}g_{-p|p}(\mathbf{x}_{-p} \mid \mathbf{x}_p)\,, \\ T_p &= \int\limits_{X_{-p}} \mathbf{t}(\mathbf{x})^T \,\mathrm{d}g_{-p|p}(\mathbf{x}_{-p} \mid \mathbf{x}_p)\,, \\ \mathbf{e} &= \mathbf{A}^{-1}\left(\mathbf{y} - \mathbf{H}\hat{\beta}\right). \end{aligned}$$

In a similar way the posterior $\mathrm{cov}^*\{E(Y \mid \mathbf{x}_p)\,,\, E(Y \mid \mathbf{x}_p)\}$ is derived, where the terms of the integrals, such as R_p and T_p can be solved analytically. At this point, for example, the expectation of $E^*(V_i) = \mathrm{var}\left[E^*\{z_i(X_i)\}\right] + E\left[\mathrm{var}^*\{z_i(X_i)\}\right]$ can be estimated. Note that the second right-hand term of $E^*(V_i)$ tend to be very small, if the design set is sufficiently large and well chosen (GP emulator is a good approximate of the real system).

2.3.5 Metabolic Pathways Representation with Graphviz

Graph visualization is a way of representing structural information of network systems as diagrams of abstract graphs and networks. Graphviz is a collection of open-source software for viewing and manipulating abstract graphs, with several main graph layout programs. Abstract directional graphs can be represented by an adjacency matrix $\mathbf{A}$, which is a $n \times n$ matrix, were n is the number of vertexes in the graph. If there is an edge from some vertex i to some vertex j, then the element $a_{i,j}$ is 1, otherwise it is zero.

Biological reaction network models are fully described by the stoichiometric matrix $\mathbf{S} \in \mathbb{R}^{n \times p}$, which elements $s_{i,j}$ are the stoichiometric coefficients. Columns correspond to reaction rates and rows correspond to compounds. The stoichiometric matrix $\mathbf{S}$ is a linear transformation of the reaction rate vector $\mathbf{r} = (r_1, \ldots, r_p)$, since

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{Sr}$$

is the system of differential equations of mass balances that characterizes all functional states $\mathbf{x} = (x_1, \ldots, x_n)$ of a reconstructed biochemical reaction network.

In order to close the mass balance, since chemical reactions cannot create or destroy elements, it should be verified that

$$\mathbf{ES} = \emptyset,$$

where $\mathbf{E}$ is the elemental matrix, which columns correspond to compounds, and rows correspond to elements, typically carbon, oxygen, nitrogen, hydrogen, phosphorous, and sulfur.

Defined the stoichiometric matrix, it is now possible to represent the dynamics of the network (reaction rate vector $\mathbf{r}$) and its structure (stoichiometric matrix $\mathbf{S}$) into a slightly modified adjacency matrix representation. If we define a vertex as a compound (or a state of the system) and the corresponding edges as mass flows, the biochemical adjacency matrix is $\hat{\mathbf{A}} = \dim(\mathbf{A})$, with n the number of compounds of the biochemical network. If there is an mass flux (edge) from some compound i to some compound j, then the element $\hat{a}_{i,j}$ is $(s_{i,j}, r_j)$, otherwise it is zero.

As an example, consider Figure 2.3.1, where the proposed long chain fatty acids (LCFA) adsorption/inhibition model from Chapter 3 is represented as a graph by the dot-layout in Graphviz. This layout is particularly suitable for drawings ”hierarchical” directed graphs, since it is designed to avoid edge crossings and reduce edge length. Because the model of the process is an open-system, three compartments are present: inflow, bio-reactor (CSTR), and products. All the edges of the graph has a label, that represent the $\hat{a}_{i,j}$ element.

In order to show the flexibility of the Graphviz visualization, the proposed LCFA adsorption/inhibition model was represented in a circo-layout (Figure 2.3.2). Compartments boundaries are lost but, on the other hand, the multiple cyclic structure of the network and the degree of connectivity between the notes (states) become more evident.

As an interesting application of this visualization tool to biochemical networks, is the possibility to visualize in time the evolution of a simulated system: if we model the vertexes as reservoirs and the edges as pipe lines, the shape size of a vertex can be associated with the compound concentration. Similarly, the thickness of an edge can be

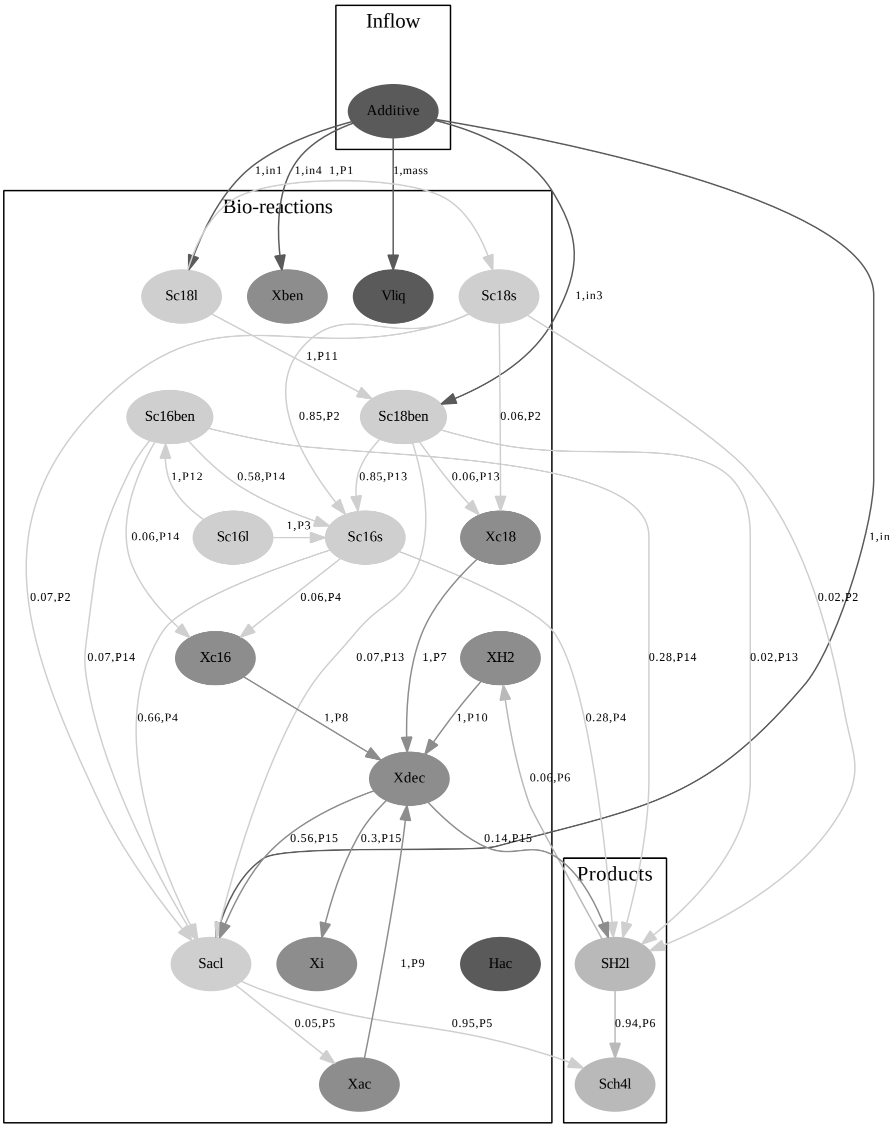


Figure 2.3.1: Graphviz-dot representation of the proposed LCFA-adsorption/inhibition model from Chapter 3.

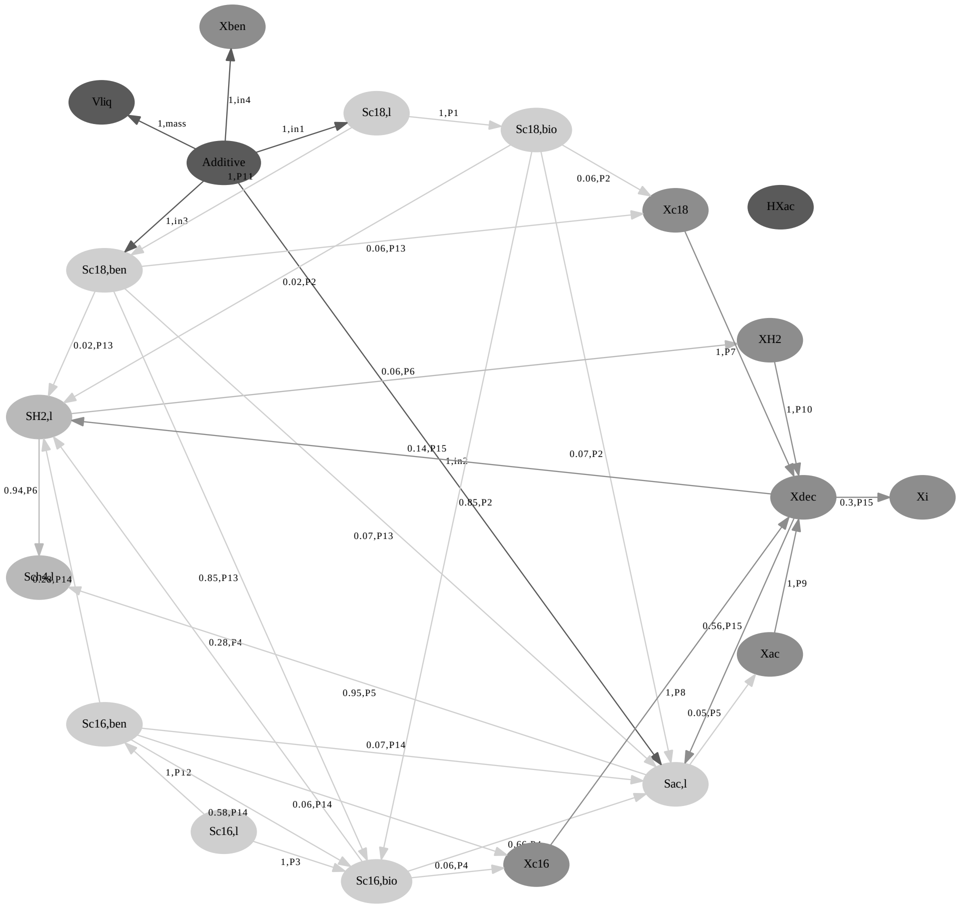


Figure 2.3.2: Graphviz *circo* representation of the proposed LCFA adsorption model from Chapter 3.

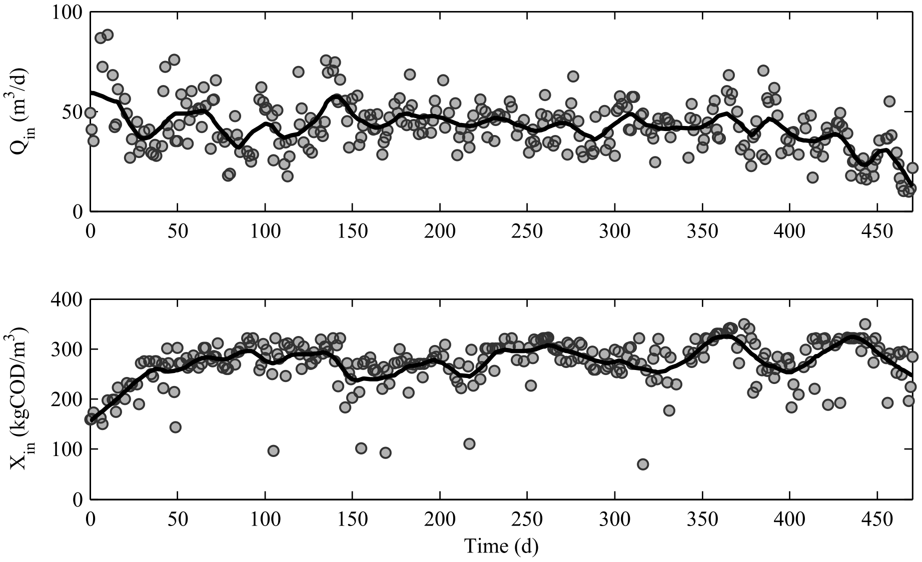


Figure 2.4.1: Pig manure mass inflow rates (Q_{in}) and inflow COD concentration (X_{in}) measurements (dots) for the biogas plant of SAVA. Data are smoothed by local regression using a first degree polynomial model weighted over a moving window of 20-days span (- line).

related to the mass flow rate. This is a simple way to visualize the overall dynamics of the system.

2.4 A Short VP-example

Consider the case of an industrial biogas plant reactor, where a simulation-model is needed for further operation analysis or to estimate missing measurements. In this short VP-example we will consider a full-scale biogas plant (SAVA, Miralcamp, Lleida, Spain), with two mesophilic AD-reactors operating in parallel. The total liquid volume (V_{liq}) is 6000 m^3 and an average hydraulic retention time (HRT) of 20 days is maintained to treat a pig manure inflow. Daily measurements are available for a period of 470 days, starting from the start-up of the plant. In particular, measurements of particulate inflow and outflow COD concentration ($X_{in/out}$), with relative mass inflow rates ($Q_{in/out}$) were collected (Figure 2.4.1). The average inflow COD concentration (X_{in}) was of 43 kgCOD/m^3, while the pH was stable (~7.8). A substrate characterization analysis confirmed that 32% of X_{in} was inert COD (results not shown, pending of publishing).

Table 2.2: Stoichiometric matrix **S** for AHM in an Excel-format.

	A	B	C	D	E	F	G
1	$\mathbf{r}/\dot{\mathbf{x}}$	*r1*	*r2*	*r3*	*r4*	*r5*	DAE
2	*Ext*		-1	-1	-1	-1	0
3	*Ssub*	-1	1				0
4	*Xbio*	Y		1			0
5	*Xi*				1		0
6	*Vliq*					1	0

Table 2.3: Reaction rate vector **r** for AHM in an Excel-format. *r1*: methanogenesis process rate; $r2$: substrate diluition rate; $r3$: biomass diluition rate; $r4$: inert diluition rate; $r5$: mass balance of the reactor.

	A	B	C
1	*r1*	mumax*Ssub/(Ssub+Ks+Ssub*Ssub/Ki)*Xbio	0
2	*r2*	D*(uSsub-Ssub)	1
3	*r3*	-D*Xbio	0
4	*r4*	D*(uXi-Xi)	1
5	*r5*	*uQin-uQout*	1

2.4.1 VP-Excel interface

A possible mathematical representation for the SAVA-biogas reactor system is by a anaerobic Haldane model (AHM), which is a simple model considering a single organic soluble substrate (S_{sub}) and a bacterial population (X_{bio}). We added one more state variable to AHM to account for the accumulation of the particulate inert COD concentration (X_{inert}) inside of the reactor. The influent particulate substrate is considered to be already fully hydrolised and the nitrogen balance is not considered. The methane production calculation is a rough estimation based on S_{sub} consumption rate. Relevant inhibition effects such as excess of free ammonia or H_2 are lumped in the Haldane inhibition term. The dynamic system is given by the stoichiometric matrix **S** and the reaction rate vector **r** described in an Excel-format in Table 2.2 and 2.5, respectively.

The state variable *Ext* in Table 2.2 is introduced because the AD-reactor is an open system, with inflow- and outflow-masses. Column-G specifies if the relative equation is of DAE-type. In our case, the AHM is described by a set of ODEs and thus, the rows relative to column-G are set to false. The reaction rates in Table 2.3 depend on a set of functions defined in Table 2.4. The "u" letter in front of a system variable (e. g., *uXsub*, *uQin*, etc.) denotes that the profile of the relative variable is known a priori (see below).

Table 2.4: AHM-functions in an Excel-format.

	A	B
1	_uSsub_	_uXsub_*(1-fi)
2	_uXi_	_uXsub_*fi
3	_Qch4_	Vliq*km*mumax*_Ssub_/(_Ssub_+Ks+pow(_Ssub_,2)/Ki)*_Xbio_
4	_D_	_uQin_/Vliq

Column-C in Table 2.3 specify if the reaction is bi-directional: for example, the mass flux of substrate (r_2) can assume positive or negative values depending on the profile of the inflow substrate COD concentration (*uSsub*). Functions from the *math* C-library are available, as for example pow($Ssub, 2$). Moreover, the table of functions (Table 2.4) is not only useful to make the reaction terms compact, but also allows to define variable stochiometric coefficients.

Information relative to model-parameters are reported in Table 2.5. Parameters are classified in three groups: reaction-rate (idx=1), stochiometric (idx=2) and initial-condition (idx=3). Providing units allows to comput a dimensional verification. Parameters that need to be estimated are defined in column-I (target) by a true/false value. In this example, we first estimated the maximum degradation rate (μ_{MAX}), the semi-saturation coefficient (K_s), the Haldane inhibition coefficient (K_i), and the initial biomass concentration of X_{bio}. Second, an alternative model of the process was assumed: the inhibition parameter K_i was set to a very high value, reducing the Haldane term to a Monod kinetic. We called such a model the anaerobic Monod model (AMM). Parameter inference was computed for AHM and the simpler AMM.

The substrate to methane ratio (k_m) was assumed perfectly known: the usual theoretical value of 0.35 m^3CH_4/kgCOD was taken. The biomass yield coefficient value (Y) of 0.1 kgCOD/kgCOD was taken from literature (de Gracia et al., 2006), while the value for the inert fraction (f_i) was provided from our own substrate characterization analysizes performed during the PSE-PROBIOGAS (2009) project. Column-E and -F in Table 2.5 define the range of definition for each parameter, which describe the uniform distribution g when global sensitivity analysis is performed. Column-G and -H define a normal prior distribution if Bayesian inference is computed. In our case the prior is uniform, since the variance of the normal distribution is infinite. Column-J identifies the parameters that are local to particular experimental designs. For example, if multiple batches are run, with different initial biomass concentrations, then the initial condition in column-D relative to *Xbio* would be written as a vector of values enclosed by squared-parenthesis.

Table 2.5: Parameters of the AHM model in an Excel-format. Column-A: idx=1 refers to the reaction rate vector; idx=2 refers to stoichiometric matrix; and idx=3 refers to initial condition values of the system. Parameters that need to be estimated are defined in column-I (target), while column-E and -F define the range of definition for each parameter. If Bayesian inference is computed then column-G and -H define a normal prior distribution. Column-J identifies the parameters that are changed during multiple experimental designs.

	A	B	C	D	E	F	G	H	I	J
1	idx	units	name	init	min	max	mu	sig	target	local
2	1	1/d	mumax	1.0	0.1	10	nan	inf	1	0
3	1	kg/m^3	Ks	5	0.1	50	nan	inf	1	0
4	1	kg/m^3	Ki	10	0.01	200	nan	inf	1	0
5	1	kg/kg	fi	0.32	0.15	0.45	0.32	0.05	0	0
6	1	m^3/m^3	km	0.35	0.1	0.8	nan	inf	0	0
7	2	kg/kg	Y	0.1	0.01	0.3	0.1	0.05	0	0
8	3	kg/m^3	*Ext*	0	-inf	inf	nan	inf	0	0
9	3	kg/m^3	*Ssub*	13.6	0	inf	nan	inf	0	0
10	3	kg/m^3	*Xbio*	1.0	0	10	nan	inf	1	0
11	3	kg/m^3	*Xi*	6.4	0	inf	nan	inf	0	0
12	3	m^3	*Vliq*	6000	6000	6000	nan	inf	0	0

This data-structure of parameters is the same as in the DRAM toolbox (Laine, 2008).

The inflow/outflow profiles of the SAVA-plant are partially reported in Table 2.6, while the measurements are given in Table 2.7. The last are used for parameter inference if this type of analysis is performed. It is possible to specify multiple experimental conditions within column-A. Measurements (e. g., *yXout*) are associated with weights (i. e. *wXout*) that specify the quality of the collected data. Missing measurements are expressed by not-a-number (nan).

Model-outputs are defined in an output-spreadsheet (Table 2.8). During parameter inference, the *yXout*-measurement (Table 2.7) are compared with the *yXout*-model output defined in Table 2.8 in order to find the MLE-value or the parameter-posterior probability distribution.

2.4.2 VP-script interface

After the AHM-structure is defined and measurements are organized in the spreadsheet, the Excel-file is uploaded to MATLAB by the following script:

```
%General Model info
Code.modelname='AHM';
```

Table 2.6: Inflow/outflow profiles in an Excel-format. Column-A is an identifier for the relative experimental design.

	A	B	C	D	E
1	#	d	kg/m^3	m^3/d	m^3/d
2	Nexp	time	*uXsub*	*uQin*	*uQout*
3	1	0	49.3	159	159
4	1	1	40.8	160	160
...	...	...	...	...	...
301	1	470	21.7	284	284

Table 2.7: Inflow/outflow measurements from the SAVA-plant in an Excel-format. Column-A is an identifier for the relative experimental design.

	A	B	C	D
1	Nexp	time	*yXout*	*wXout*
2	1	0	14.9	1
3	1	1	16.0	1
4	1	2	nan	1
...	...	...	...	...
302	1	470	17.5	1

Table 2.8: AHM-simulation outputs as defined in an Excel-table format.

	A	B	C
1	*ySsub*	*Ssub*	kg/m^3
2	*yXbio*	*Xbio*	kg/m^3
3	*yXi*	*Xi*	kg/m^3
4	*yQch4*	*Qch4*	m^3/d
5	*yXout*	*Xbio+Xi+Ssub*	kg/m^3

```
Code.infoauthor={'Zivko␣Juznic-Zonta';['Copyrigth,' date];...
                 'Mollet␣del␣Valles,␣BCN,␣Spain'; 'UPC/GIRO-CT'};
Code.infomodel={'Modified␣Haldane␣model␣for␣AD␣-␣SAVA␣plant'};
%Load the model from Excel
S=loadxls(Code);
%Transform in a symbolic representation and prepare for MCMC-sampling
S=sysinfo(S); S=mcmcparam(S);
```

The information relative to AHM is stored in the data-structure S. For example, the differential equation relative to the soluble substrate COD concentration is simply recovered by typing

```
>> dSsub_dt=S.fullODE(2,1)
dSsub_dt = -(Ssub-uSsub)*D-(Ssub*Xbio*mu)/(Ks+Ssub+pow(Ssub,2)/Ki)
```

When the symbolic toolbox of MATLAB is available, data-structure S contains the sensitivity trajectory function for each state/parameter combination. For example, if we are interested in how S_{sub} change with a local change of μ_{MAX}, then the following command returns the sensitivity trajectory function $\mathrm{d}Ssub/\mathrm{d}\mu_{MAX}$:

```
>> dSsub_dmumax=S.SensFunc(1,2)
dSsub_dmumax = -(Ssub*Xbio)/(Ks+Ssub+pow(Ssub,2)/Ki)
```

The adjacency matrix **A** for AHM is returned by

```
%Path network diagram (.dot file for Graphviz)and the adjacency matrix
Adj_s=pathnet(S,Code)
Adj_s =
     4     1     1     1     1
     0     2     1     0     0
     0     0     2     0     0
     0     0     0     1     0
     0     0     0     0     1
```

where at the same time the corresponding Graphviz-code (i. e. the file AHM.dot) is generated. The relative Graphviz-diagram is represented in Figure 2.4.2. Note how the bi-directional arrows evidence the fact that the reaction rate r_2, r_4 and r_5 may be positive or negative, since there is an exchange of mass between the external environment (*Ext* state) and the AD-reactor. The only internal process of the AD-reactor is the bio-reaction r_1.

The script below builds the data-structure C (`buildCode`), which contains all the necessary information to build the Cmex-code (or MATLAB-code) is saved in the AHM.c file. This file is then compiled (`CmexModel`) to build the AHM.mex32 file that is used inside an S-function Simulink block. Finally the data-class PNM (Pathway Network Model) is built to perform simulations.

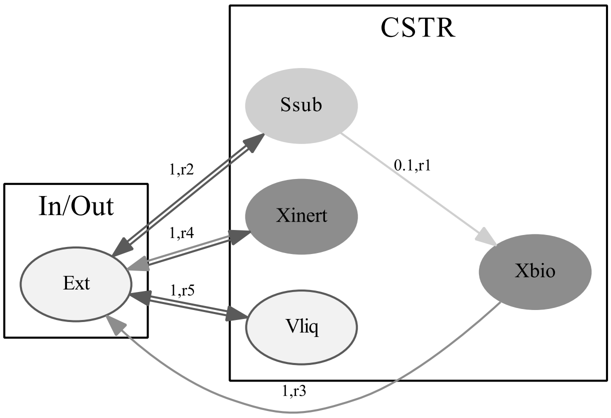


Figure 2.4.2: AHM-structure graph by Graphviz-dot representation.

```
%Build the Code structure
Code=buildCode(S,Code);
%Build and compile the Cmex-code for the S-function (Simulink)
CmexModel(Code,true);
%...or M-code for SUNDIALS solver
%MfileModel(Code,'sundials',S.xls.DAE);
%Generete the data class of the model
PNM=modelclass(S,Code);
```

Before performing parameter inference, it is useful to perform a global SA of the defined goodness-of-fit measure. In our case, the mean square error (MSE) was used, but any likelihood function could be considered. The framework for sensitivity analysis in model-calibration is described in detail by Ratto et al. (2001) and in the following Chapter 3 and 4. Briefly, by applying a global SA to goodness-of-fit measure, parameters driving model runs with good fit-to-data are identified. Moreover, parameter interaction features are highlighted. Because the sensitivity indexes are calculated by the GEM-SA tool, we first need to provide the model input/output training data to build a GP emulator. We run the model over 256 parameter-samples taken from a latin hypercube design, where 20% of sampels were used for cross-validation of the GP emulator. The following script was executed to build the input/output txt-file for the GEM-SA tool:

```
%Input file for GEM-SA
str={'\mu_{MAX}' 'K_s' 'K_i' 'X_{bio}(0)'};
[param_lhs,MSE_lhs]=SAsslhs(PNM, 256, str);
```

The MSE measure relative to each parameter-sample for AHM is represented in Figure 2.4.3. Parameters μ_{MAX} and K_s have a high influence over the MSE variance and thus, it is expected that they have a better chance of reducing the variance of the

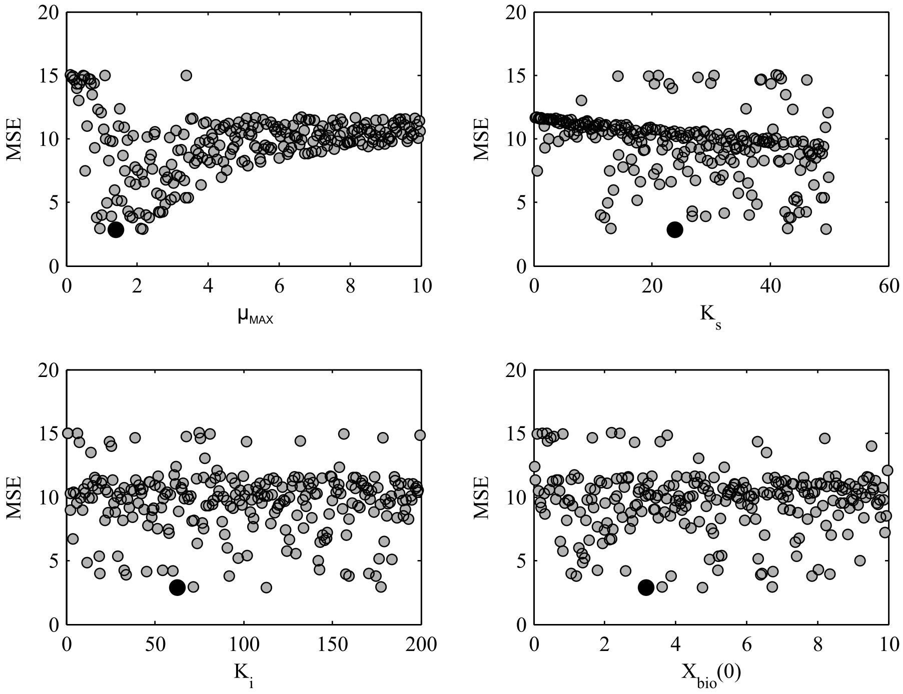


Figure 2.4.3: Mean square error (MSE) relative to the latin hypercube sample of AHM-parameters. The black-filled point identifies the minimum MSE-value for the given LHS-design.

Table 2.9: SA sensitivity indexes for AHM and AMM model-parameters. S_i: main effect index, S_{Ti}: total effect index. Units: %.

	AHM-parameters, θ_i				AMM-parameters, θ_i		
$S_{i,j}$	μ_{MAX}	K_s	$X_{bio}(0)$	K_i	μ_{MAX}	K_s	$X_{bio}(0)$
K_s	30.1				32.14		
$X_{bio}(0)$	1.13	0.43			0.94	0.04	
K_i	2.68	0.65	0.09				
S_i	39.3	16.2	0.12	0.28	56.65	9.59	0.44
S_{Ti}	82.1	55.8	7.7	8.6	89.93	41.97	1.62

MSE value. In particular, μ_{MAX} has a strong non-monotonic relation with MSE and expresses the majority of the MSE variance: this suggests that its precision of estimated is relatively high. We suspect that K_i and $X_{bio}(0)$ are unidentifiable since they cannot influence directly the MSE-value. It is impossible to make any statement regarding the interaction effects between parameters if only the 1D-scatter-plots are investigated. The same procedure was repeated for the AMM-model and very similar scatter-plots to Figure 2.4.3 were found.

The results of SA, obtained from the GEM-SA tool, are given in Table 2.9 for the AHM- and the AMM-model. The analysis confirms that μ_{MAX} and K_s can be considered practically identifiable because of their significant main effect indexes (S_i). On the other hand, K_i and $X_{bio}(0)$ are unidentifiable since they cannot influence alone (i. e. through main effects) the MSE measure. The main effect index $S_{X_{bio}(0)}$ is low because $X_{bio}(0)$ influences the system only during the start-up period of the SAVA-plant operation. The model-structure of AHM should provide insight over a possible inhibition process, but since K_i is unidentifiable the AHM-model looses its explanatory advantage over AMM.

When AHM is considered, the main effects alone expresses only 55.9% of the total MSE-variance (Table 2.9), which means that a larger reduction in variance may be achieved if one could identify the interacting parameters. Large differences between S_i and S_{Ti} are a sign of over-parametrization. For both models, μ_{MAX} is the most influential parameter (high S_{Ti} value) and the first-order interaction effect between μ_{MAX} and K_s (S_{μ_{MAX},K_s}) is the most important. This implies that μ_{MAX} and K_s are very needed for a good fit-to-data. The sum of main and first-order interactions for AHM is of 90.1%, which means that the remaining 9.9% of variance is due to higher-order effects. On the other hand, for the AMM-model, all the MSE-variance (99.8%) is explained by the main effect and first-order interactions. Thus, we expect that the correlation structure of AHM would be more complex (i. e. nonlinear) than for AMM.

Table 2.10: MLE-parameter values.

Parameters, θ_i	AHM	AMM
μ_{MAX}	2.49	0.86
K_s	48.65	12.17
$X_{bio}(0)$	4.26	3.94
K_i	15.86	(inf)
σ	2.81	2.80

The above SA gives quantitative and synthetic information about the parameter-structure after conditioning on data, but gives no information about the most probable parameter-estimate or the behavioural regions of the parameters. In order to obtain this "topological" information of the parameter-structure the following script-code computes the maximum likelihood estimate (MLE) and the parameter posterior probability distribution (Bayesian inference):

```
%Calibration of parameters (MLE) with scatter-search routine
ResultsMLE=optimizationGO(PNM);
%Bayesian inference with the DRAM-sampler
ResultsPOST=optimizationMCMC(PNM);
```

The MLE-parameter values estimated by the SS-optimization routine for AHM and AMM are reported in Table 2.10. Note that for both models the estimated measurement standard deviation error σ (i. e. MSE) is approximately the same. The extra functional-flexibility that the inhibition term provides is not useful in this model-calibration scenario. Thus, based on the Occam's razor criteria we should prefer AMM.

Because the estimation of MLE involves a maximization problem of a likelihood function, it is regarded as a simple problem if compared with the multi-dimensional integration problem of finding the mean estimate of the posterior-parameter distribution. However, in our case, the likelihood function is multi-modal and that converts the estimation of MLE in a difficult optimization problem. In practice, the SS-optimization algorithm has many internal parameters that should be specified with care to achieve satisfactory results. Moreover, because of time-discretization errors and parameter identifiability issues finding the MLE may be impossible. The time-discretization errors can be resolved by setting the relative/absolute error of the ODE/DAE solver to a very low value, but this implies that time per simulation-run increases considerably. We preferred to set a reasonable ODE-solver precision and permit a noisy objective surface, but this choice forced us to disabled the local searches during the SS-optimization. However, above we found that some parameters are unidentifiable and thus, the surface of the

Table 2.11: Simple statistics of the posterior distribution MCMC-samples for AHM and AMM. Geweke's diagnostic measure Geweke (1992) is used to verify the convergence of the MCMC-chains.

AHM-model

Parameter, θ_i	mean	std	MCMC error	median	2.5%	97.5%	Geweke
μ_{MAX}	1.51	0.51	0.044	1.33	0.89	2.56	0.91
K_s	25.66	10.67	0.821	22.50	11.68	47.59	0.90
K_i	73.92	57.82	4.729	54.38	10.98	191.57	0.87
$X_{bio}(0)$	4.08	0.51	0.026	4.05	3.21	5.20	0.98
σ	2.80	0.12	0.001	2.80	2.59	3.04	0.99

AMM-model

Parameter, θ_i	mean	std	MCMC error	median	2.5%	97.5%	Geweke
μ_{MAX}	0.94	0.15	0.004	0.90	0.74	1.31	0.98
K_s	14.24	4.37	0.119	13.23	8.57	25.50	0.97
$X_{bio}(0)$	4.15	0.50	0.019	4.13	3.31	5.28	0.99
σ	2.81	0.11	0.004	2.56	2.81	3.04	0.99

likelihood is flat in those parameter-directions: in such case there is no guarantee of convergence.

The above problems are easily handled by the Bayesian procedure. Bayesian inference via MCMC has a theoretic guarantee than the MCMC algorithm will converge if run long enough. In our case, the total chain length was of 65,000, with a burn-in period of 5,000 and a post-burn-in of 5,000 MCMC-samples. Convergence was verified by the Geweke's diagnostic measure Geweke (1992). Summarizing statistics of the parameter-posterior distribution parameters for AHM and AMM are given in Table 2.11. A comparison of the MLE-values from Table 2.10 with the mean value from Table 2.11 suggests that the AHM-posterir distribution is heavily asymmetric, while the AMM-posterior is just slighly asymmetric. It is well known that when the posterior is asymmetric the mode (i. e. MLE) is the poorest choice of centrality measure and the mean may be more appropriate. High difference between the mode and the mean could indicate that the AHM-posterior is multi-modal. We decomposed the AHM-posterior distribution as a mixture of Gaussian distributions (McLachlan and Peel, 2000) as suggested by Carreira-Perpiñán (2000) and found that the AHM-posterior was bi-modal. Briefly, a Gaussian mixture model (GMM) was build on the MCMC-samples and only high-probability components were retained. Two high-probability regions were found for the AHM-posterior.

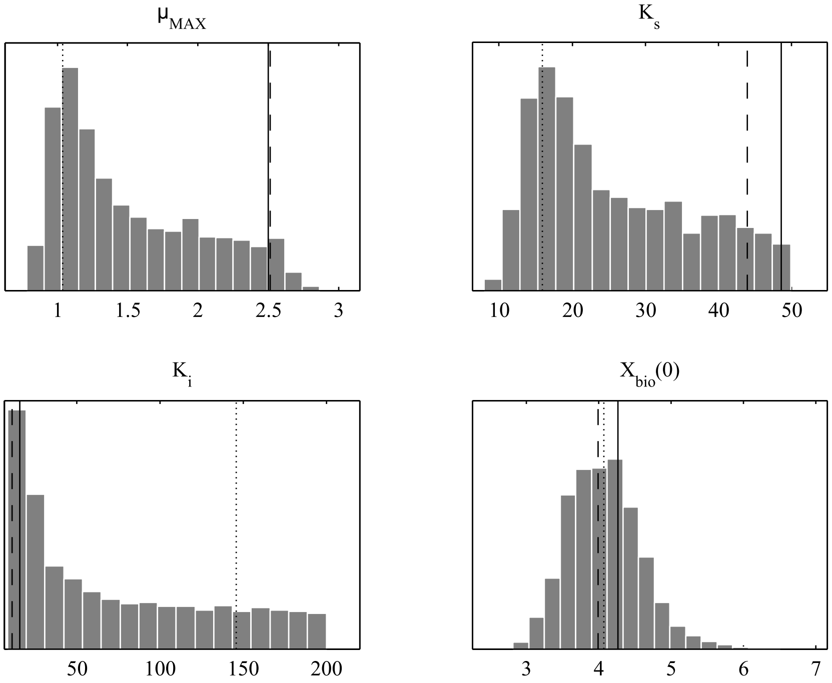


Figure 2.4.4: Histograms for the relative marginal posterior distributions of the AHM-parameters. MLE are estimated from SS-optimization (- line), GMM-approximation of the posterior (- - line) and sub-optimal mode of GMM (: line).

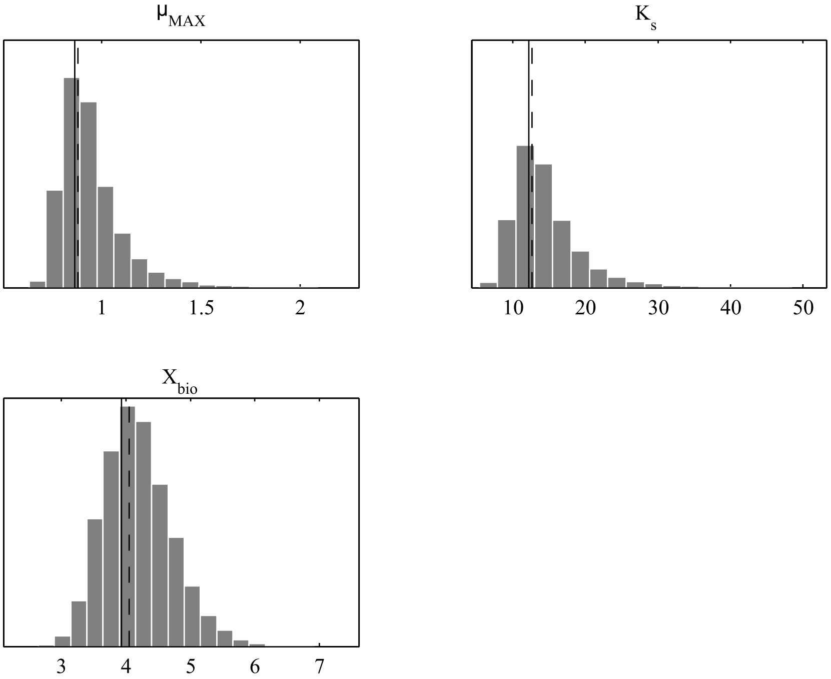


Figure 2.4.5: Histograms for the relative marginal posterior distributions of the AMM-parameters. MLE are estimated from SS-optimization (- line) and GMM-approximation of the posterior (- - line).

Marginal posterior distributions relative to AMH are represented in Figure 2.4.4. The relative MLE-values were also reported. The highest mode (- - line in Figure 2.4.4) was approximately the MLE found by the SS-optimization routine (- line in Figure 2.4.4). The highest mode was associated with the GMM-component that accounted for $\sim$40% of the entire posterior mass. On the other hand, the sub-optimal mode (: line in Figure 2.4.4) was associated with the renaming 60% of the AHM-posterior mass. Thus, the usual recommendation to choose the highest mode could be misleading since the highest mode is uncharacteristic of the majority of the posterior. Choosing the global or the sub-optimal MLE value would lead to very different conclusions about the process operation since their relative values are quite different. In this case, we are in doubt which mode to consider, not only to interpret model-parameters but even to build frequentist confidence regions. When multi-modality is an issue, it may simply result impossible to characterize parameter-precision if the frequentist procedure is selected.

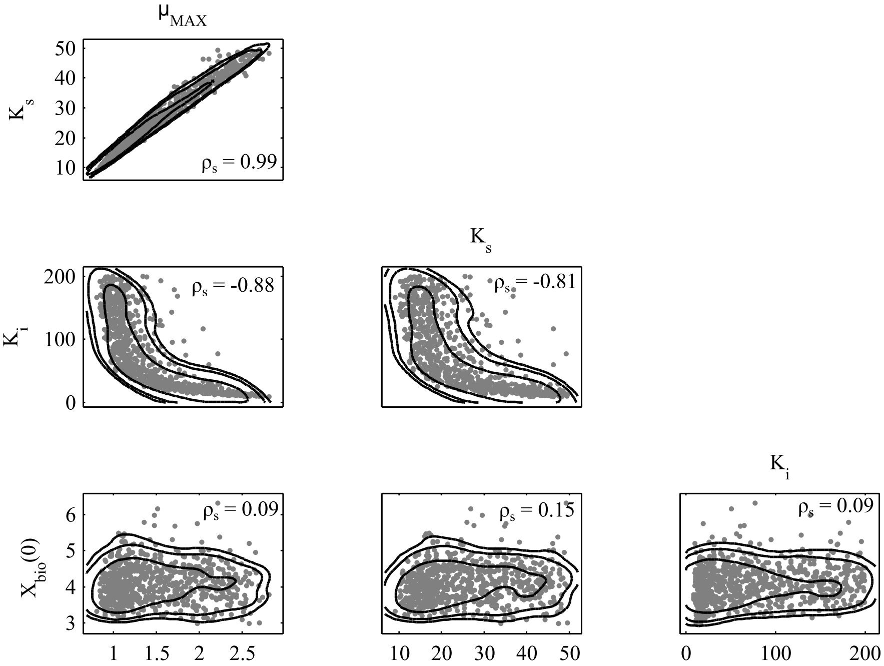


Figure 2.4.6: Pairs of MCMC-samples of AHM-parameters with a 2D-kernel estimation contours (62%, 90% and 95%) and Spearman's rank correlation coefficient.

In Figure 2.4.5, the marginal posterior distributions relative to AMM are represented. The AMM-posterior is uni-modal and slightly asymmetric. In this case, the mean and particularly the median are more robust summary statistics of centrality then the mode. In a similar way, the inter-quartile range is more robust then the standard deviation to describe the variation of the distribution. If we imagine that uniform distribution component is subtracted from the AMH-posterior, then the marginal AMM-posterior distributions (Figure 2.4.5) is similar to the AHM's (Figure 2.4.4). This uniform distribution is mainly associated with the nonidentifiability of K_i. The marginal posterior distribution associated with $X_{bio}(0)$ is preserved. Even if the marginal posterior distribution of $X_{bio}(0)$ is quite narrow, we remark that the relative main effect index $S_{X_{bio}(0)}$ is very low. This implies that the marginal of $X_{bio}(0)$ may be very sensible to the particular choice of other parameters rather then data. Note that the MLE of AMM-parameters (Figure 2.4.5) is closer to the sub-optimal MLE of AHM-parameters (Figure 2.4.4). This result confirms the importance of the above "sub-optimal" GMM-component.

Correlation analysis of the posterior joint distribution of AHM-parameters is represented in Figure 2.4.6. The posterior distribution reveals a strong linear correlation between μ_{MAX} and K_s. Note, that from SA analysis we expected an interaction between

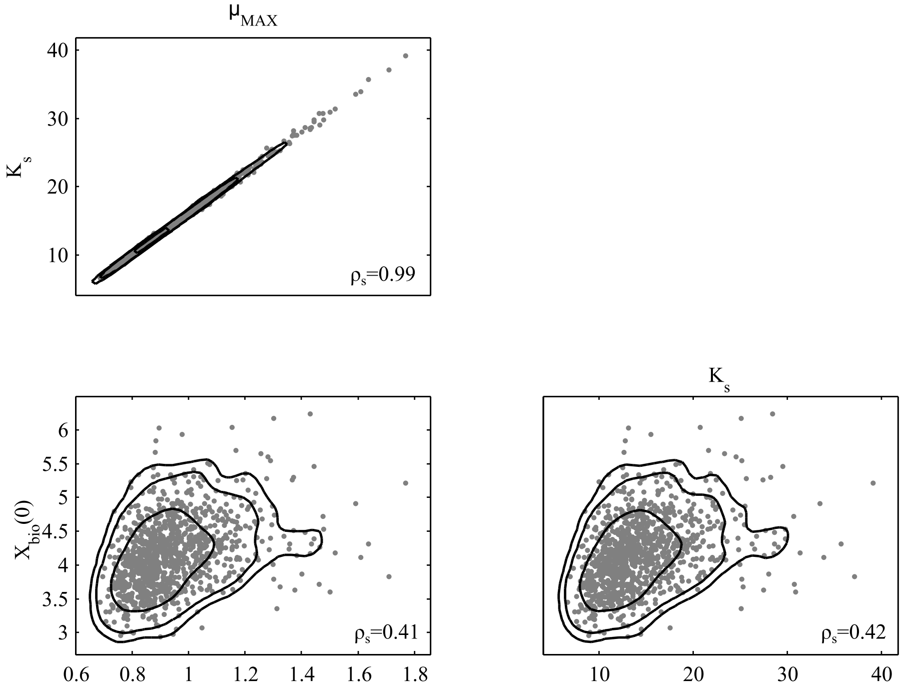


Figure 2.4.7: Pairs of MCMC-samples of AMM-parameters with a 2D-kernel estimation contours (62%, 90% and 95%) and Spearman's rank correlation coefficient.

μ_{MAX} and K_s, but only analyzing the "topology" of the posterior (i. e. the likelihood in our case) we could determine the structure of such interaction. Nonlinear correlations are observed for μ_{MAX} vs K_i and K_s vs K_i interactions, while $X_{bio}(0)$ is not involved in any first-order interaction. The banana-shape of the posterior joint distribution and the strong correlation between parameters reveals why convergence to MLE-value is difficult for gradient-based optimization algorithms.

Correlation analysis of the posterior joint distribution of AMM-parameters is represented in Figure 2.4.7. Monod-term parameters, μ_{MAX} and K_S, are still very correlated. In contrast with the AHM-case, we observe that correlation between $X_{bio}(0)$ and the Monod-prameters is significant. However, the influence of this correlation on the MSE-measure is marginal because the first-order interaction sensitivity index (S_{ij}) relative to $X_{bio}(0)$ is very low. When data are sparse relative to model-complexity, taking a simpler model may simplify the analysis of the parameter-posterior. In the case of AMM-model, the multi-modal parameter-posterior present in the AHM-case disappears.

Even if the AMM-model is still over-parametrized (i. e. high value of $S_{Ti} - S_i$) the parameter-structure is useful to estimate missing measurements or hidden states of the system. In particular, the methane production and the total biomass concentration of

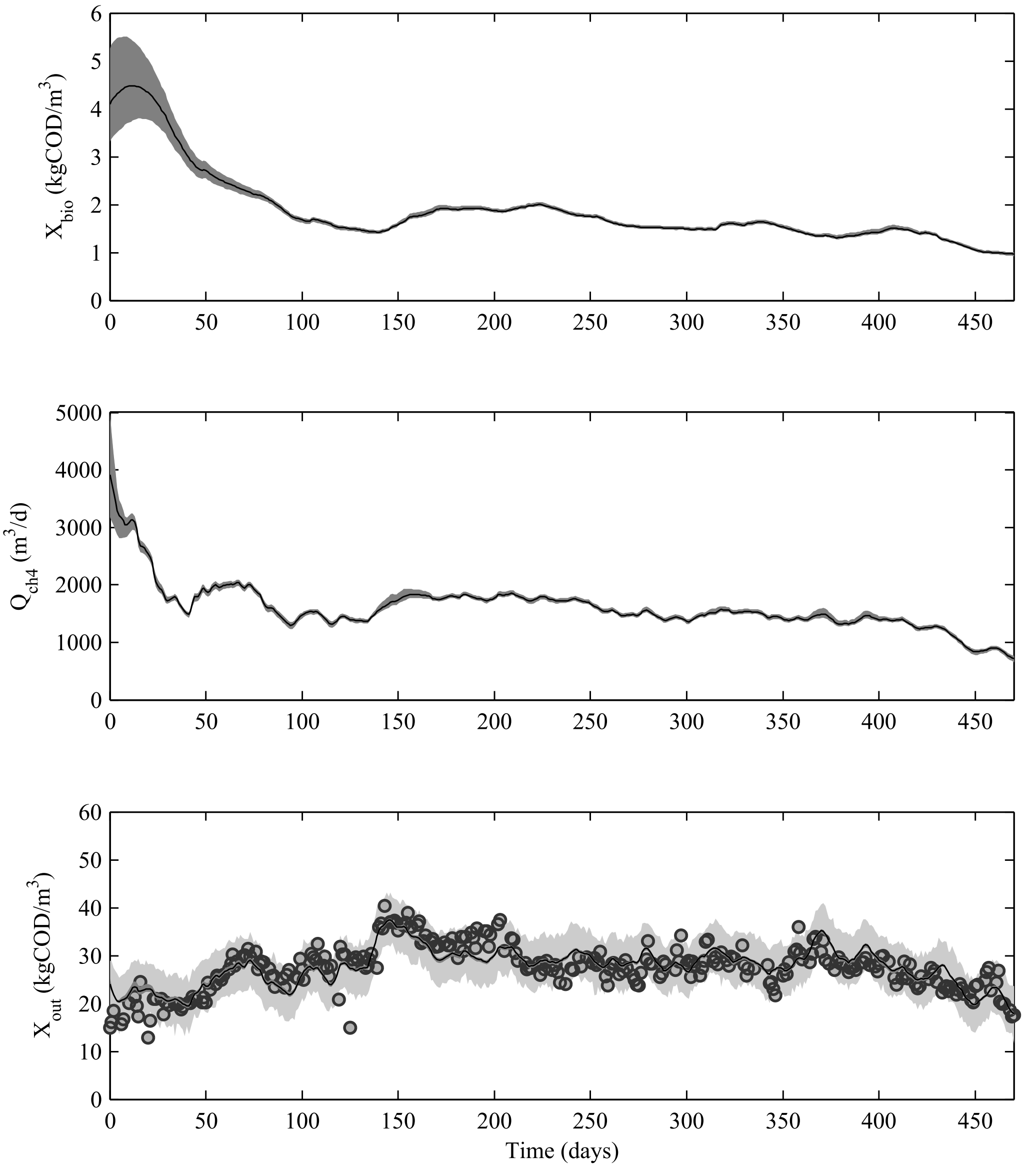


Figure 2.4.8: AHM prediction of methane production, Q_{ch4}, and total COD concentration at the outlet, X_{out}, for the SAVA biogas plant. The gray envelop is the parameter prediction uncertainty envelop (95% credible interval), and the light-gray envelop is the measurement prediction uncertainty (95% credible interval).

the SAVA biogas plant can be estimated conditional on the above inferred parameter-uncertainty. In Figure 2.4.8 the parameter prediction uncertainty envelop at 95% (gray) is represented for the biomass concentration (X_{bio}), methane production (Q_{ch4}), and total COD outflow concentration (X_{out}). The light-gray envelop is the measurement prediction uncertainty. The measurement prediction uncertainty envelop is not available for Q_{ch4} because measurements for methane production were not available. Even if the model cannot be used for extrapolation because of its highly correlated parameters, the parameter-uncertainty structure do not compromise its filtering ability over X_{tot}. The uncertainty of X_{bio} at time zero is equal to the estimated parameter-uncertainty of $X_{bio}(0)$ and it considerably decreases as the system reaches its semi-steady-state. The small variance of X_{bio} over the entire simulation is because perfect knowledge over the value of the biomass yield (Y) was assumed. The parameter prediction uncertainty envelops for X_{tot} and Q_{ch4} are quite tiny. This is because methane production estimation is based on the assumption that the substrate to methane ratio, k_m, was considered perfectly known. However, if we do not agree with this assumption we could characterize k_m (or Y) as a random variable, described by a suitable probability distribution (or prior) that represents our knowledge over the possible values of k_m. Then, the parameter prediction uncertainty envelop for the methane production can be recomputed conditional on k_m-uncertainty. One advantage of Bayesian inference is that it provides the input for uncertainty analysis, which is conditioned not only on prior expert's knowledge but even on collected data.

It is evident from Figure 2.4.8 that residuals are auto-correlated. We left this crucial observation for the end to highlight its importance. In fact, in order to achieve a reliable information on parameter uncertainty the statistical model assumes that residuals should be independent and identically distributed (IID). A very simple empirical approach to account for auto-correlation in residuals is data whitening (Dochain and Vanrolleghem, 2001; Daebel, 2006). Pre-withening procedure proposed by Daebel (2006) was applied to meet statistical IID assumptions. Results for parameter uncertainty estimation relative to AMM are presented in Table 2.12. For uncorrelated residuals, the parameter uncertainty increased more than two times (e. g. inter-quartile ranges) if compared with the results relative to auto-correlated residuals (Table 2.11). Auto-correlation of the residuals leads to underestimation of parameter errors. Thus, evaluation of parameter uncertainty is reliable only if the relative statistical model is shown correct.

The following script allows to estimate the error covariance matrix within the frequentist procedure.

```
% Estimated mode value of the model-parameters if likemisfit
```

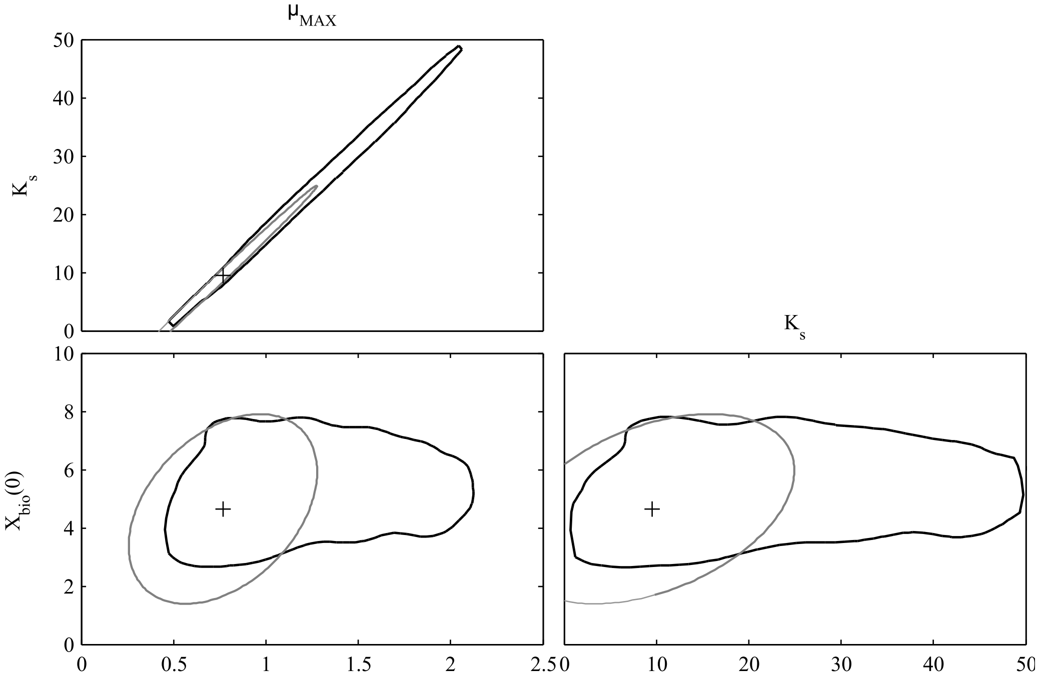


Figure 2.4.9: 2D-kernel estimation contours (- line) and frequentist confidence ellipses (- gray line) relative to AMM with pre-withening.

Table 2.12: Simple statistics of the posterior distribution MCMC-samples for AMM within a pre-whitening of data.

AMM-model (Data whitening with addition of normal noise, $Norm(0,\ 5^2)$)

Parameter, θ_i	mean	std	MCMC error	median	2.5%	97.5%	Geweke
μ_{MAX}	1.07	0.35	0.020	0.96	0.66	2.09	0.98
K_s	18.72	10.51	0.119	15.27	6.40	45.84	0.94
$X_{bio}(0)$	5.02	1.05	0.019	4.86	3.40	7.58	0.92
σ	6.17	0.25	0.008	6.18	5.70	6.66	0.99

```
x_mode=ResultsMLE.xbest;
% Misfit function is a negative normal log-likelihood
likemisfit=true;
J=@(x)ssobjectiveGO(x,PNM,likemisfit);
% Estimate the Hessian (D'Errico, 2006)
H_est=hessianest(J,x_mode);
% Covariance matrix estimate
Ch_est=2*inv(H_est);
 % Confidence intervals at 95%
delta=sqrt(diag(Ch_est)) * tinv(1-0.05/2,v);
ci=[(x_mode(:)-delta) (x_mode(:) + delta)];
```

For AMM with pre-withening, Figure 2.4.9 represents the frequentist confidence ellipses (- gray line) and the Bayesian MCMC-posterior estimate (- line in Figure 2.4.9). The linear approximation underestimates the parameter-uncertainty because of the strong asymmetric shape of the posterior. In this case, the linear approximation provided by the frequentist procedure was found unreliable to describe parameter uncertainty. However, the covariance matrix estimated from the Hessian could be found still useful: remark that the DRAM-sampler adapt the proposal covariance matrix using the chain generated during its HM-steps. The frequentist covariance matrix could be used as an initial candidate for the proposal covariance matrix of the DRAM-sampler in order to speed-up convergence of the chain.

Acknowledgements

To Ministry of Economy and Competitivity, for financial suport to the Spanish Singular and Strategic Project PROBIOGAS(PSE-120000-2008-10).

Bibliography

Cariboni, J., Gatelli, D., Liska, R., Saltelli, A., 2007. The role of sensitivity analysis in ecological modelling. Ecological Modelling 203 (1-2), 167 – 182, special Issue on Ecological Informatics: Biologically-Inspired Machine Learning, 4th Conference of the International Society for Ecological Informatics.

Carreira-Perpiñán, M. A., 2000. Mode-finding for mixtures of gaussian distributions. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (11), 1318–1323.

Checchi, N., Giusti, E., Marsili-Libelli, S., 2007. PEAS: A toolbox to assess the accuracy of estimated parameters in environmental models. Environmental Modelling & Software 22 (6), 899–913.

Checchi, N., Marsili-Libelli, S., 2005. Reliability of parameter estimation in respirometric models. Water Research 39 (15), 3686–3696.

Daebel, H., 2006. Parameter uncetainties in modeling urban wastewater systems. Ph.D. thesis, University of Karlsruhe (TH), Swiss Federal Institute of Technology Zurich.

de Gracia, M., Sancho, L., García-Heras, J. L., Vanrolleghem, P., Ayesa, E., 2006. Mass and charge conservation check in dynamic models: application to the new ADM1 model. Water Science and Technology 53 (1), 225–240.

D'Errico, J., 2006. Adaptive robust numerical differentiation, matlab file exchange (accessed November 2011).
URL http://www.mathworks.com/matlabcentral/fileexchange/13490-adaptive-robust-numerical-differentiation

Dochain, D., Vanrolleghem, P. A., 2001. Dynamical Modelling and Estimation in Wastewater Treatment Processes. IWA Publishing, UK.

Geweke, J., 1992. Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. In Bayesian Statistics 4 (ed. Bernado, J. M., Berger, J. O., Dawid, A. P., Smith, A. F. M.). Oxford University Press, UK.

Houska, B., Ferreau, H. J., Diehl, M., 2011. ACADO toolkit. an open-source framework for automatic control and dynamic optimization. Optimal Control Applications and Methods 32 (3), 298–312.

Kennedy, M., 2004. Description of the gaussian process model used in GEM-SA. Tech. rep., University of Sheffield.

Kennedy, M., O'Hagan, A., 2001. Bayesian calibration of computer models. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63 (3), 425–464.

Laine, M., 2008. Adaptive MCMC methods with applications in environmental and geophysical models. Ph.D. thesis, Department Of Mathematics And Physics; Lappeenranta University Of Technology Lappeenranta.

Laine, M., Tamminen, J., 2008. Aerosol model selection and uncertainty modelling by adaptive MCMC technique. Atmospheric Chemistry and Physics 8 (24), 7697–7707.

Larrosa, J. A. E., 2008. New heuristics for global optimization of complex bioprocesses. Ph.D. thesis, Departamento de Enxeneria Quimica; Universidade de Vigo.

Marsili-Libelli, S., Guerrizio, S., Checchi, N., 2003. Confidence regions of estimated parameters for ecological systems. Ecological Modelling 165 (2-3), 127–146.

McLachlan, G., Peel, D., 2000. Finite Mixture Models. John Wiley and Sons, USA.

Oakley, J. E., O'Hagan, A., 2004. Probabilistic sensitivity analysis of complex models: a Bayesian approach. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66 (3), 751–769.

Pauw, D. D., 2005. Optimal experimental design for calibration of bioprocess models: a validated software toolbox. Ph.D. thesis, Faculteit Bio-ingenieurswetenschappen; University of Gent.

Petersen, B., Gernaey, K., Vanrolleghem, P. A., 2001. Practical identifiability of model parameters by combined respirometric-titrimetric measurements. Water Science and Technology 43 (7), 347–355.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., Flannery, B. P., 1997. Numerical recipes in C : the art of scientific computing. Cambridge University Press.

PSE-PROBIOGAS, 2009. Manual de estado del arte de la co-digestión anaerobia de residuos ganaderos y agroindustriales. Tech. rep., GIRO-CT, Mollet del Vallès, Barcelona.

Ratto, M., Tarantola, S., Saltelli, A., 2001. Sensitivity analysis in model calibration: GSA-GLUE approach. Computer Physics Communications 136 (3), 212–224.

Reichert, P., von Schulthess, R., Wild, D., 1995. The use of aquasim for estimating parameters of activated sludge models. Water Science and Technology 31 (2), 135–147.

Reichl, G., 2003. Wastewater - a library for modeling and simulation of wastewater treatment plants in modelica. In: Fritzson, P. (Ed.), Proceedings of the 3rd International Modelica Conference, Linköping.

Rodriguez-Fernandez, M., Egea, J. A., Banga, J. R., 2006a. Novel metaheuristic for parameter estimation in nonlinear dynamic biological systems. BMC Bioinformatics 7, 483.

Saltelli, A., Ratto, M., Tarantola, S., Campolongo, F., 2006. Sensitivity analysis practices: Strategies for model-based inference. Reliability Engineering & System Safety 91 (10-11), 1109–1125.

Seber, G. A. F., Wild, C. J., 1989. Nonlinear Regression. John Wiley & Sons Ltd., USA.

Vrugt, J. A., Bouten, W., 2002. Validity of first-order approximations to describe parameter uncertainty in soil hydrologic models. Soil Science Society of America Journal 66, 1740–1751.

3 Modelling Inhibitory Effect of LCFA in the Anaerobic Digestion Process

Title: Modelling inhibitory effect of long chain fatty acids in the anaerobic digestion process (submited to Water Research)

Živko J. Zonta[b], Maria Magdalena Alves[c], Xavier Flotats[a,b]and Jordi Palatsi[a,*]

[a]GIRO Technological Centre. Rambla Pompeu Fabra 1, 08100 Mollet del Vallès, Barcelona, Spain
[b]Department of Agrifood Engineering and Biotechnology. Universitat Politècnica de Catalunya. Campus del Baix Llobregat, Edifici D4, Esteve Terradas 8, 08860 Castelldefels, Barcelona, Spain
[c]Institute for Biotechnology and Bioengineering, Centre of Biological Engineering, University of Minho, Campus de Gualtar, 4710-57 Braga, Portugal.
[*]Corresponding author

3.1 Abstract

Causal modeling of anaerobic digestion within the International Water Association Anaerobic Digestion Model N$^{\text{o}}$1 (ADM1) framework was used to answer a retrospective question regarding causes of long chain fatty acids (LCFA) inhibition. Data was obtained from methanogenic activity tests, batch-assays at different initial LCFA concentration and batch-assays including bentonite addition, using two different granular anaerobic biomasses. New kinetics were considered to describe the bio-physics of the inhibitory process: i) adsorption of LCFA over granular biomass and ii) specific LCFA-degrader populations. Furthermore, a new state variable was introduced to describe the state of damage of the acetoclastic population, in order to account for a loss of cell-functionality induced by the adsorbed LCFAs. A comparison was performed between models that use a conventional non-competitive inhibition function and the proposed a "healthy-state" description for the LCFA-inhibition process. The model-parameter practical identifiably was assessed by global sensitivity analysis. Calibration and model structure validation were performed on independent data sets. The importance of microbial population structure (saturated/in-saturated LCFA-degraders) was evidenced for a successful degradation process. We show that, under the ADM1 framework, reliable simulations of the LCFA-inhibition process can be achieved if the model includes the description of the adsorptive nature of the LCFAs and the LCFA-damage over the biomass.

3.2 Introduction

Despite the fact that LCFA-inhibition is well documented and has a significant impact on the anaerobic digestion process, this phenomenon has still not been included in ADM1 reference model (Bastone et al., 2002). In other developed models, LCFA inhibition is mainly modeled as a non-competitive process on the lipolytic, acetogenic or methanogenic activities (Angelidaki et al., 1999; Salminen et al., 2000; Lokshina et al., 2003). However, the physical adsorption of LCFA and its inhibition process or the microbiological aspects of LCFA-degradation remain poorly characterized.

Up today, Hwu et al. (1998) have proposed one of the most detailed descriptions of the LCFA's bio-sorption, degradation and inhibition processes. The description is based on a four-phase theoretical conceptual model. First, after a LCFA-pulse, LCFA rapidly disappears from the aqueous phase and adsorbs onto the solid phase. Because of the LCFA-toxicity effect, the methane production is negligible. Second, depending on the initial LCFA-pulse concentration, an LCFA-concentration increase is detected in the aqueous phase, because of a biologically mediated desorption. Third, the LCFA-concentration decrease in the aqueous phase because of a biological degradation of the adsorbed LCFA. Finally, the methane formation is recovered when the remaining LCFA-adsorbed concentration is low.

The cell-membrane seems to be the prime common target for most of the described LCFA-inhibition experiences According to Kim and Gadd (2008), cell-membrane exposure to high concentrations of LCFA promotes macromolecular crowding and disruption of mechanisms such as proton-motive-force, DNA-docking and ATP-synthesis. More recently, Pereira et al. (2004) and Pereira et al. (2005) proven that LCFA-inhibition was reversible and was related to physical transport limitation effects. The irreversible cell-damage, due to the adsorption of LCFA was discarded after this evidence and new technological perspectives emerged for the high-rate treatment of anaerobic wastewater with lipids (Alves et al., 2007). Also impairment of nutrient uptake or inhibition of specific enzyme activity were reported (Desbois and Smith, 2010). Archaea favor active maintenance over survival modes because they are adapted to thrive with chronic energy stress. Methanogens, for example, can adapt in several ways the structure and dynamics of their membranes (Valentine, 2007). The commonly used non-competitive inhibition functions (Angelidaki et al., 1999; Palatsi et al., 2010) implicitly assume that, after a LCFA-shock, the time to restore cell-functionality is negligible. Consequently, those classical models may result inappropriate to simulate heavily LCFA-inhibited systems.

A number of studies have discussed the addition of competing adsorbents into systems treating grease and fats (Angelidaki et al., 1999; Beccari et al., 1999; Nielsen and Ahring, 2006; Palatsi et al., 2009) as a possible strategy to limit LCFA inhibitory effects. However, solid-liquid adsorption dynamics were not included in those studies, and thus, only approximations to LCFA-adsorption (ratio inhibitor/biomass) were considered for modeling purposes (Pereira et al., 2004; Palatsi et al., 2010). To our knowledge, a mathematical model that includes adsorption-inhibition-degradation processes has not yet been tested.

Recent advances in the molecular microbial ecology have brought new insights on the specific microorganisms that are involved in the *ß*-oxidation process and the syntrophic methanogens interactions (Hatamoto et al., 2007; Sousa et al., 2007). Those microorganisms are not always abundant in non-adapted systems and their dynamics are difficult to follow. In this context, mathematical models can be used as a valuable tool to interpret collected data and test hypothesis.

This chapter aims to propose a LCFA-adsorption-inhibition sub-model that can be integrated into the ADM1-model. Unknown model-parameters are inferred within two independent data sets obtained from previous batch experiments (Palatsi et al., 2012). A simple LCFA inhibition-adsorption model is compared with a slightly more complex model that tries to represent the damage of the acetoclastic population after a LCFA overload. Biomass structure is analyzed and its implications are discussed.

3.3 Material and Methods

3.3.1 Experimental Observations

The experimental set-up consisted on several specific batch tests performed with two different anaerobic granular sludges (sludge-A and sludge-B) and with bentonite as a clay-mineral adsorbent. The experimental set-up is extensively described in Palatsi et al. (2012). Here, the experimental observations obtained in Palatsi et al. (2012) were grouped in three main data-sets, summarized as follows:

Data set D_1: LCFA-adsorption, batch-test with chemically inactivated biomass (sludge-A) and bentonite; monitoring the soluble-LCFA time evolution ($\mathrm{LCFA_l}$).

Data set D_2: Sluge methanogenic activity test (SMA) with sludge-A ($\mathrm{D_{2,A}}$) and sludge-B ($\mathrm{D_{2,B}}$) to acetate (Ac) and hydrogen (H_2) as biogas formation substrates; monitoring the accumulated methane production (CH_4). In addition, blank assays (vials with biomass but without added substrates) were performed for sludge-A and sludge-B.

Data set D_3: Batch-tests with increasing LCFA-concentration and batch-tests with specific prevention/recovering strategies where bentonite was introduced as an exogenous LCFA adsorbent. The experiments for sludge-A ($\mathrm{D_{3,A}}$) included vials with bentonite addition after a LCFA-pulse (T_A vials). The experiments for sludge-B ($\mathrm{D_{3,B}}$) included vials with bentonite-LCFA mixed compound added to LCFA-free biomass (T_B vials). Control vials with LCFA but without bentonite (C_A and C_B vials) were also considered. Concentration-measurements were taken for solid-LCFA ($\mathrm{LCFA_s}$), liquid-LCFA ($\mathrm{LCFA_l}$), volatile fatty acids (VFA) and methane production (CH_4).

3.3.2 Models Development

The developed models were based on a simplification of the anaerobic process as described by the ADM1 model. The same structure, nomenclature and units of the ADM1 model were used (Bastone et al., 2002). The first proposed model, LCFA-M1, included a LCFA-adsorption process and non-competitive inhibition functions. The second model, LCFA-M2, also included a new state variable called *healthy-state* that considers the LCFA-inhibitory stage of biomass. The models were implemented in MATLAB (The Mathworks, USA) within the Simulink Cmex-coded environment.

For the proposed LCFA-inhibition models, the differential equations for the LCFA-soluble matter in the liquid phase are:

$$\frac{\mathrm{d}S_{c18,l}}{\mathrm{d}t} = -\mathrm{P}_1 - \mathrm{P}_2 \tag{3.3.1}$$

$$\frac{\mathrm{d}S_{c16,l}}{\mathrm{d}t} = -\mathrm{P}_5 - \mathrm{P}_6 \tag{3.3.2}$$

$$+\mathrm{P}_5 + (1 - Y_{fa})\,(1 - \beta_{ac} - \beta_{h2})\,\mathrm{P}_7 - \mathrm{P}_8, \tag{3.3.3}$$

while in the solid phase are:

$$\frac{\mathrm{d}S_{c18,s}}{\mathrm{d}t} = \mathrm{P}_1 + (1 - Y_{fa})\,(1 - \beta_{ac} - \beta_{h2})\,\mathrm{P}_3 - \mathrm{P}_4 \tag{3.3.4}$$

$$\frac{\mathrm{d}S_{c18,ben}}{\mathrm{d}t} = \mathrm{P}_2 - \mathrm{P}_3 \tag{3.3.5}$$

$$\frac{\mathrm{d}S_{c16,s}}{\mathrm{d}t} = (1 - Y_{fa})\,(1 - \beta_{ac} - \beta_{h2})\,\mathrm{P}_4 + $$

$$+\mathrm{P}_5 + (1 - Y_{fa})\,(1 - \beta_{ac} - \beta_{h2})\,\mathrm{P}_7 - \mathrm{P}_8 \tag{3.3.6}$$

$$\frac{\mathrm{d}S_{c16,ben}}{\mathrm{d}t} = \mathrm{P}_6 - \mathrm{P}_7. \tag{3.3.7}$$

Differential equations for acetate, hydrogen andCH_4 in the liquid phase are:

$$\frac{\mathrm{d}S_{ac,l}}{\mathrm{d}t} = (1 - Y_{fa})\,\beta_{ac}\mathrm{P}_3 + (1 - Y_{fa})\,\beta_{ac}\mathrm{P}_4 + (1 - Y_{fa})\,\beta_{ac}\mathrm{P}_7 + \tag{3.3.8}$$

$$+\,(1 - \alpha)\,(1 - Y_{fa})\,\mathrm{P}_8 - \mathrm{P}_9 + 0.8\,(1 - f_{Xi})\,\mathrm{P}_{15} \tag{3.3.9}$$

$$\frac{\mathrm{d}S_{h2,l}}{\mathrm{d}t} = (1 - Y_{fa})\,\beta_{h2}\mathrm{P}_3 + (1 - Y_{fa})\,\beta_{h2}\mathrm{P}_4 + (1 - Y_{fa})\,\beta_{h2}\mathrm{P}_7 + \tag{3.3.10}$$

$$+\alpha\,(1 - Y_{fa})\,\mathrm{P}_8 - \mathrm{P}_{10} + 0.2\,(1 - f_{Xi})\,\mathrm{P}_{15} \tag{3.3.11}$$

$$\frac{\mathrm{d}S_{ch4,l}}{\mathrm{d}t} = (1 - Y_{ac})\,\mathrm{P}_9 + (1 - Y_{h2})\,\mathrm{P}_{10}, \tag{3.3.12}$$

while for the particulate matter are:

$$
\begin{aligned}
\frac{\mathrm{d}X_{c18}}{\mathrm{d}t} &= Y_{fa}\mathrm{P}_3 + Y_{fa}\mathrm{P}_4 - \mathrm{P}_{11} && (3.3.13)\\
\frac{\mathrm{d}X_{c16}}{\mathrm{d}t} &= Y_{fa}\mathrm{P}_7 + Y_{fa}\mathrm{P}_8 - \mathrm{P}_{12} && (3.3.14)\\
\frac{\mathrm{d}X_{ac}}{\mathrm{d}t} &= Y_{ac}\mathrm{P}_9 - \mathrm{P}_{13} && (3.3.15)\\
\frac{\mathrm{d}X_{h2}}{\mathrm{d}t} &= Y_{ac}\mathrm{P}_{10} - \mathrm{P}_{14} && (3.3.16)\\
\frac{\mathrm{d}X_{dec}}{\mathrm{d}t} &= \mathrm{P}_{11} + \mathrm{P}_{12} + \mathrm{P}_{13} + \mathrm{P}_{14} - \mathrm{P}_{15} && (3.3.17)\\
\frac{\mathrm{d}X_{i}}{\mathrm{d}t} &= f_{Xi}\mathrm{P}_{15}. && (3.3.18)
\end{aligned}
$$

Next, we report the relative process(es) expressed in kg COD m^{-3} d^{-1} for $S_{c18,l}$ (C18 = oleate) adsorption over biomass

$$P_1 = k_{ads,bio} \cdot S_{c18,l} \cdot (q_{sat,bio} \cdot X_{bio} - S_{c18,s}) - k_{des,bio} \cdot S_{c18,s}, \quad (3.3.19)$$

$S_{c18,l}$ adsorption over bentonite

$$P_2 = k_{ads,ben} \cdot S_{c18,l} \cdot (q_{sat,ben} \cdot X_{ben} - S_{c18,ben}) - k_{des,ben} \cdot S_{c18,ben}, \quad (3.3.20)$$

$S_{c18,ben}$ biological desorption from bentonite

$$P_3 = k_{m,fa} \cdot \frac{S_{c18,ben}}{K_{S,fa} + S_{c18,ben}} \cdot X_{c18} \cdot I_{h2} \cdot I_{Xfa}, \quad (3.3.21)$$

$S_{c18,s}$ degradation

$$P_4 = k_{m,fa} \cdot \frac{S_{c18,bio}}{K_{S,fa} + S_{c18,bio}} \cdot X_{c18} \cdot I_{h2} \cdot I_{Xfa}, \quad (3.3.22)$$

$S_{c16,l}$ (C16 = palmitate) adsorption over biomass

$$P_5 = k_{ads,bio} \cdot S_{c16,l} \cdot (q_{sat,bio} \cdot X_{bio} - S_{c16,s}) - k_{des,ben} \cdot S_{c18,ben}, \quad (3.3.23)$$

$S_{c16,l}$ adsorption over bentonite

$$P_6 = k_{ads,ben} \cdot S_{c16,l} \cdot (q_{sat,ben} \cdot X_{ben} - S_{c16,ben}) - k_{des,ben} \cdot S_{c16,ben}, \quad (3.3.24)$$

$S_{c16,ben}$ biological desorption from bentonite

$$P_7 = k_{m,fa} \cdot \frac{S_{c16,ben}}{K_{S,fa} + S_{c16,ben}} \cdot X_{c16} \cdot I_{h2} \cdot I_{Xfa}, \tag{3.3.25}$$

$S_{c16,s}$ degradation

$$P_8 = k_{m,fa} \cdot \frac{S_{c16,bio}}{K_{S,fa} + S_{c16,bio}} \cdot X_{c16} \cdot I_{h2} \cdot I_{Xfa}, \tag{3.3.26}$$

S_{ac} degradation

$$P_9 = k_{m,ac} \cdot \frac{S_{ac}}{K_{S,ac} + S_{ac}} \cdot X_{ac} \cdot I_{Xac}, \tag{3.3.27}$$

S_{h2} degradation

$$P_{10} = k_{m,h2} \cdot \frac{S_{h2}}{K_{S,h2} + S_{h2}} \cdot X_{h2}, \tag{3.3.28}$$

biomass decay

$$\begin{aligned} P_{11} &= k_{dec} \cdot X_{c18}, & (3.3.29)\\ P_{12} &= k_{dec} \cdot X_{c16}, & (3.3.30)\\ \mathrm{P}_{13} &= k_{dec} \cdot X_{ac}, & (3.3.31)\\ P_{14} &= k_{dec} \cdot X_{h2}, & (3.3.32) \end{aligned}$$

and X_{dec} slowly-biodegradable COD recirculation

$$P_{15} = k_{hyd} \cdot X_{dec} \cdot I_{Xfa}, \tag{3.3.33}$$

where the inhibition functions I will be defined in the following, since they constitute the main contribution of this work.

In order to clarify the meaning of the above equations, a scheme of the simplified anaerobic digestion model is presented in Figure 3.3.1, based on the following assumptions:

- Only LCFA (S_{fa}), acetate (S_{ac}), hydrogen (S_{h2}) and methane (S_{ch4}) were considered as the main model components in order to keep the structure of the model simple. No other substrates as lipids (X_{li}), proteins (X_{pr}) and carbohydrates (X_{ch}) were considered. In accordance with the experimental results (Palatsi et al., 2012),

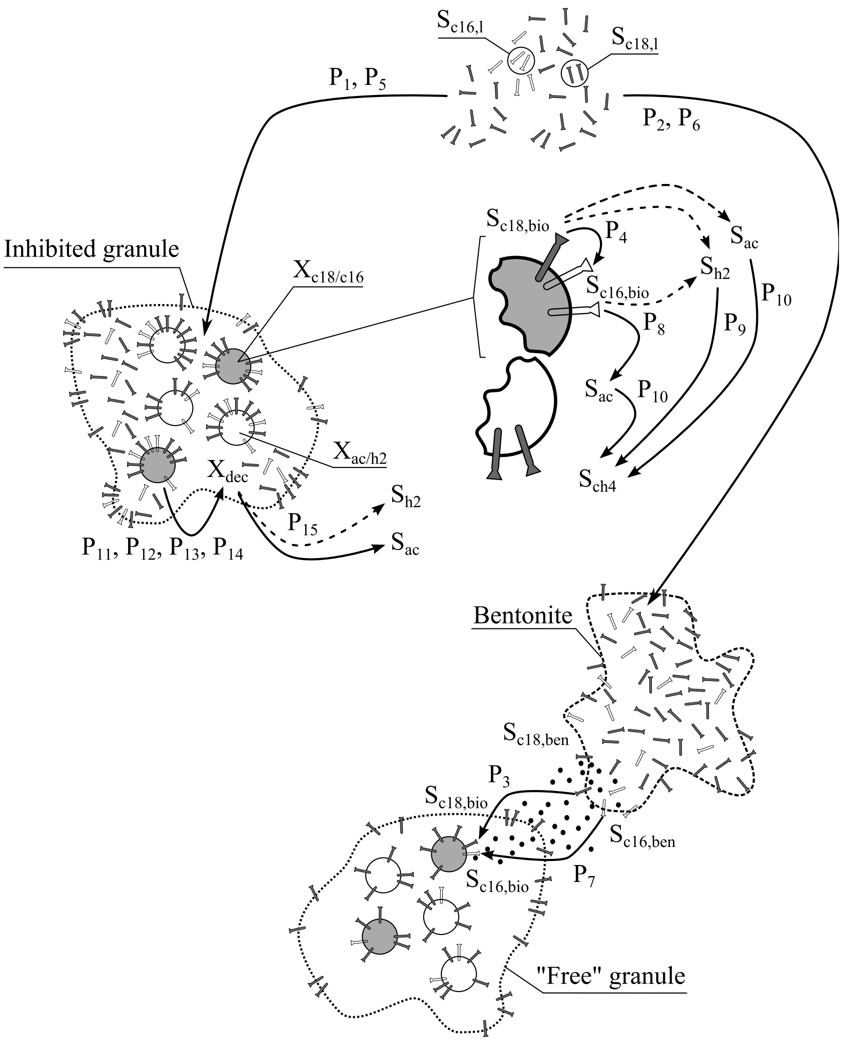


Figure 3.3.1: Process scheme of the assumed LCFA-adsorption and degradation pathway with/without clay mineral (bentonite) addition as exogenous adsorbent. The process P_i is represented, where $X_{c18/c16}$ are the oleate/palmitate degraders, $X_{ac/h2}$ are the methanogens and X_{dec} is the decayed biomass and the considered slowly bio-degradable substrate. The LCFA-substrates are the oleate/palmitate present in the liquid ($S_{c18/c16,l}$), adsorbed on biomass ($S_{c18/c16,bio}$) and on bentonite ($S_{c18/c16,ben}$).

butyrate (S_{bu}), valerate (S_{va}) and propionate (S_{pro}) were not considered as they are in the ADM1 model. Thus, $S_{ac}$and S_{h2} were the only products of the ß-oxidation process of LCFA (P$_4$ and P$_8$ in Eq. (3.3.22) and Eq. (3.3.26), respectively). Sodium oleate (S_{c18}) was assumed as the main substrate. Particulate decayed biomass $X_{dec}$was considered as a storage for all the slowly biodegradable-substrates. $X_{dec}$was estimated for each experimental design by the COD mass balance of the system. It was assumed that 1 gCOD of X_{dec} is converted through hydrolysis to 0.58 gCOD of acetate (S_{ac}), 0.14 gCOD of hydrogen (S_{h2}) and 0.30 gCOD of inerts (X_i). A first-order rate (k_{hyd}) was assumed for the hydrolysis process of X_{dec} (P$_{15}$ in Eq. (3.3.33)).

- The total LCFA concentration S_{fa} was split into oleate S_{c18} and palmitate S_{c16} since palmitate has been proposed to be the main intermediate during the anaerobic degradation of oleate (Lalman and Bagley, 2001; Pereira et al., 2002). Oleate and palmitate can be found free in liquid media ($S_{c18,l}$ or $S_{c16,l}$) or adsorbed onto biomass ($S_{c18,bio}$ or $S_{c16,bio}$) and bentonite ($S_{c18,ben}$ or $S_{c16,ben}$), when this clay-mineral is added to the media as an exogenous adsorbent. Moreover, during oleate degradation (P$_4$), palmitate was accumulated onto biomass as suggested before by Pereira et al. (2002) and Palatsi et al. (2012). According to Hwu et al. (1998), the LCFA adsorption onto anaerobic biomass ($S_{fa,s}$) is described as a pre-requisite for its biological degradation (P$_1$ and P$_5$ in Eq. (3.3.19) and Eq. (3.3.23), respectively). The process of LCFA-adsorption over bentonite was also considered (P$_2$ and P$_6$ in Eq. (3.3.20) and Eq. (3.3.24), respectively) when bentonite was added in the system. The biological-mediated desorption was not considered in the adsorption-model as proposed by Hwu et al. (1998). For simplicity, the desorption from solid to liquid was assumed dependend only on the adsorbed LCFA-concentration, $S_{fa,s}$. The liquid-solid transport dynamics were approximated by a *Langmuir* adsorption isotherm kinetic (Mouneimne et al., 2004), which was expressed by the following differential equation form:

$$\frac{\mathrm{d}S_{fa,s}}{\mathrm{d}t} = k_{ads}S_{fa,l}\left(q_{sat}X_{ads} - S_{fa,s}\right) - k_{des}S_{fa,s} \tag{3.3.34}$$

where $S_{fa,s}$ and $S_{fa,l}$ are respectively the LCFA concentration in the solid and liquid phase, k_{ads} is the adsorption rate, k_{des} is the desorption rate, X_{ads} is the adsorbent concentration, and q_{sat} is the adsorbante over adsorbent saturation coefficient. The considered adsorbents (X_{ads}) were bentonite (X_{ben}) and granular sludge (X_{bio}). The notation of LCFA concentration adsorbed only on bentonite (or biomass) is $S_{fa,ben}$ (or $S_{fa,bio}$), while LCFA adsorbed over all the present solids is written as $S_{fa,s}$. Adsorption interactions effects between multiple components that are present in the liquid-solid system were not considered in the adsorption-model. The concentration of the overall biomass-adsorbent X_{bio} was considered time-variable since it is the sum of specific substrate-degraders (e. g., X_{fa}, X_{ac}, X_{h2}, etc.), inerts (X_i) and the slowly-biodegradable substrates (X_{dec}). On the other hand, X_{ben} was assumed constant when it was used.

- Only one *ß*-oxidation step was considered (process P_3 and P_7 in Eq. (3.3.21) and Eq. (3.3.25), respectively) in order to model the transference of the adsorbed LCFA on bentonite ($S_{fa,ben}$) to biomass ($S_{fa,bio}$). An exo-enzymatic action was assumed to be mediated by the LCFA-degraders since they may grow on the outmost shell of the granule (Picioreanu et al., 2005) in direct contact with the surface of bentonite.

- Two different groups of specific LCFA-degraders microorganisms (X_{fa}) were considered: i) the oleate-degraders, X_{c18}, and ii) the palmitate-degraders, X_{c16}. Sousa et al. (2008) reported that oleate/palmitate-degrading cultures showed a different microbial composition, concluding that the community structure in a reactor might depend on the saturation degree of the LCFA-feed. This result suggests that not all the *ß*-oxidative degraders have the ability to degrade saturated (e. g., S_{c18}) and unsaturated (e. g., S_{c16}) fatty acids. Thus, specific LCFA-degraders microorganisms are needed to process a saturated or unsaturated LCFA-substrate.

- A non-competitive inhibition function of LCFA over *ß*-oxidazing population (X_{c18} or X_{c16}) was considered, defined as

$$I_{Xfa} = K_{Xfa} \left(K_{Xfa} + S_{fa,bio}\right)^{-1} , \qquad (3.3.35)$$

where K_{Xfa} is the inhibitory concentration coefficient and $S_{fa,bio}$ is the adsorbed LCFA onto biomass We assumed that only $S_{fa,bio}$ causes LCFA-inhibition since other possible LCFA-species as $S_{fa,l}$ or $S_{fa,ben}$ are not involved in the disruption of the cell-functionality. The non-competitive LCFA-inhibition function I_{Xfa} was also considered as an inhibitory function for the hydrolysis process as suggested by Angelidaki et al. (1999).

- According to Hanaki et al. (1981), the aceticlastic methanogens (process P_{10} in Eq. (3.3.28)) are probably the most LCFA-affected microorganisms. Thus, for the aceticlastic population, we assumed a secondary non-competitive LCFA-inhibition function (Salminen et al., 2000; Lokshina et al., 2003), defined as

$$I_{Xac} = I_{Xac,noncomp} = K_{Xfa} \left(K_{Xac} + S_{fa,bio}\right)^{-1}, \tag{3.3.36}$$

where K_{Xac} is the corresponding inhibitory concentration coefficient. Here, the non-competitive LCFA-inhibition function I_{Xac} -Eq. (3.3.36)- was used in the first proposed version of the LCFA-inhibition model (LCFA-M1). In the following, we propose a second model for the LCFA-inhibition process of the aceticlastic population (LCFA-M2). We introduce a new state variable, H_{Xac}, called *healthy-state* of the aceticlastic population X_{ac}, which is defined as

$$\frac{\mathrm{d}H_{Xac}}{\mathrm{d}t} = r_{max} \cdot (1 - H_{Xac}) - S_{fa,bio} \cdot H_{Xac}, \tag{3.3.37}$$

where r_{max} is the maximum cell recovery rate. The healthy state H_{Xac} is defined within a finite range [0, 1]: i) if H_{Xac} is one then the average functionality of the cell-membrane is optimal (methanogenic patway is on); while ii) if H_{Xac} is zero then the cell-membrane is severely damaged and the methanogenic pathway is interrupted to other cell-maintenance/recovery pathways. The cell-damage, D_{Xac}, can be quantified as D_{Xac} = 1 - H_{Xac}. The rate of recovery depends on the level of damage of the cell (first term on the right-hand side of Eq. (3.3.37)): if the cell is highly damaged then the recovery rate is maximal. The rate of the damage (second term on the right-hand side of Eq. (3.3.37)) depends directly on the present value of $S_{fa,bio}$ and H_{Xac}: if the LCFA-adsorbed concentration on the biomass is high then the damage to the cell is high. However, if H_{Xac} is almost zero then no further damage is possible. We assumed that under extreme environmental pressure (rich-LCFA concentrations) the acetoclastic population becomes more resilient to LCFA-damages because of its biochemical adaptation (Valentine, 2007; Kim and Gadd, 2008) and its increased effort to restore the cell-functionality (i. e. increase the recovery rate). When the healthy-state is zero (H_{Xac} = 0) it does not mean that biomass has reached a state of decay. In the present study, the rate of decay of the biomass is independent of H_{Xac}.

We assumed that an acetoclastic microorganism switches from a survival-mode to a metanogenic-mode only when its cell-functionality is restored to a given level. The LCFA-inhibition function is assumed smooth since it is an average measure of the overall acetoclastic population transition from the survival to the normal functionality mode. Thus, the LCFA-inhibition function for the acetoclastic population is a continuous function, defined as:

$$I_{Xac}the = I_{Xac,healthy} = H_{Xac}^{\gamma}, \tag{3.3.38}$$

where γ is the state of health coefficient and is defined over the interval $[1, +\infty)$. Note that after a LCFA-shock for a value of $\gamma > 1$, the recovery of the methanogenic activity is fully re-activated only when the average cell-damage is considerably reduced. Because H_{Xac} is defined in a finite range [0, 1], $I_{Xac,healthy}$ takes values in finite range [0, 1].

Summarizing, the diference between LCFA-M1 and LCFA-M2 is only in how the inhibition function I_{Xac} is defined: LCFA-M1 is characterized by $I_{Xac,noncomp}$ defined in Eq. (3.3.36), while LCFA-M2 is characterized by $I_{Xac,healthy}$ defined by Eq. (3.3.38).

- Contrary to other proposed models (Palatsi et al., 2012), no LCFA-inhibition effect was considered for the hydrogenotrophic methanogens (process P_9 in Eq. (3.3.27)). This choice is supported by the experimental evidence from activity tests over an LCFA-adsorbed (inhibited) biomass (Pereira et al., 2003). The activity tests with hydrogen, acetate, propionate and butyrate indicated a positive activity only for the vial fed with H_2 (Pereira et al., 2003). LCFA-inhibited biomass, which was fed respectively with acetate, propionate and butyrate, activated the methane production only when the adsorbed LCFA was completely depleted. Pereira et al. (2003) suggested that the LCFA-adsorbed layer on the membrane of the methanogenic-degraders hindered the transport of the substrates from the bulking liquid onto the cells. The authors suggested that the diffusion of H_2 through the LCFA-layer was faster than for the other substrates, because its molecular weight was very low. Thus, even if an inhibitory concentration of LCFA was adsorbed over the biomass, vials fed with the H_2-substrate immediately transformed this substrate into methane.

- We assumed the following non-competitive inhibition function over the LCFA-degraders population

$$I_{h2} = K_{I,h2}\left(K_{I,h2} + S_{fa,bio}\right)^{-1}, \tag{3.3.39}$$

where $K_{I,h2}$ is the corresponding inhibitory concentration coefficient, in order to account for the effect of a possible high partial pressure of hydrogen (Bastone et al., 2002).

3.3.3 Practical Identification and Global Sensitivity Analysis

Environmental models are known to contain ill-defined or non-identifiable parameters. When the same mathematical model is calibrated, practical non-identifiablity of parameters depends not only on the model-structure, but even on the evidence D (available data) with which it is compared. Parameter practical identifiably can be precisely assessed within a global sensitivity analysis (SA) by studying how model-parameters affect a misfit function, J. Performing a SA of J, involves the decomposition of its variance over the parameter-space. *Variance-based* methods (Sobol', 1976) are well suited to account for the parameter interactions when non-linear models are considered (Saltelli et al., 2010). A variance-based main effect for a generic parameter θ_i (i = 1,. . . , k) can be written as

$$V_{\theta_i}\left(E_{\theta_{\sim i}}\left\{J\,|\,\theta_i\right\}\right), \tag{3.3.40}$$

where θ_i is the i-th parameter and $\theta_{\sim i}$ denotes the matrix of all parameters but θ_i. The meaning of the inner expectation operator, E, is that the mean of J is taken over all possible values of $\theta_{\sim i}$ while keeping θ_i fixed. The outer variance, V, is taken over all the possible values of θ_i. When the main effect is normalized by the unconditional variance, V(J), we obtain the associated sensitivity measure (main effect index, S_i) written as (Saltelli et al., 2010)

$$S_i = \frac{V_{\theta_i}\left(E_{\theta_{\sim i}}\left\{J\,|\,\theta_i\right\}\right)}{V\left(J\right)}. \tag{3.3.41}$$

In a similar way, the first-order interaction effect index ($S_{i,j}$) can be written as

$$S_{i,j} = \frac{V_{\theta_{i,j}}\left(E_{\theta_{\sim i,j}}\left\{J\,|\,\theta_{i,j}\right\}\right)}{V\left(J\right)}. \tag{3.3.42}$$

Another popular variance based measure is the total effect index (S_{Ti}), defined as

$$S_{Ti} = \frac{E_{\theta_{\sim i}}\left(V_{\theta_i}\left\{J\,|\,\theta_{\sim i}\right\}\right)}{V\left(J\right)} = 1 - \frac{V_{X_{\sim i}}\left(E_{\theta_i}\left\{J\,|\,\theta_{\sim i}\right\}\right)}{V\left(J\right)}, \tag{3.3.43}$$

which measures the first and higher order effects (interactions) of the parameter θ_i. In probabilistic SA, the parameter θ is a stochastic variable characterized by a distribution $g(\theta)$ that describes our prior assumptions over θ. In the present work, two types of uncertainty parameter distributions $g(\theta)$ with respective parameters a and b were used as needed: a uniform distribution, $Unif(a,b)$, and a normal distribution, $Norm(a,b^2)$. When $g(\theta_i)$ was of uniform-type, during model-calibration (i. e. least-square function J minimization), the parameter θ_i was constrained over a finite range interval given the relative uniform parameter interval $[\ a_i,\ b_i\ ]$, while when it was of normal-type, θ was constraint positive with a six-sigma (i. e. $6\times b_i$) variation around its location parameter, a_i.

Provided with the above sensitivity measures, Ratto et al. (2001) proposed general guidelines to asses the practical identifiability of model-parameters: i) parameters with a high main effect (high S_i) affect J singularly, independent of interactions and thus can be considered precisely estimated; ii) parameters with a small main (S_i) and total effect (S_{Ti}) have a negligible effect over J and thus cannot be estimated precisely; iii) parameters with a small main effect (S_i) but high total effect (S_{Ti}) affect J mainly through interactions.

In our case, we used a sum-of-squares misfit function $J(\theta;\mathrm{D})$. Weights relative to the number of samples and measurement-errors were not applied. Because the number of CH_4 samples was very high in relation to other measurements (e. g., Ac, $LCFA_s$ and $LCFA_l$) we implicitly prioritized the fit to the methane production samples. The SA was performed by a Bayesian sensitivity analysis tool for estimating the main, first-order and total effect indexes (Oakley and O'Hagan, 2004).

3.3.4 Sequential Model Calibration

The model-parameter, least-squares (LS) estimate of θ was computed within a "scatter-search" global optimization routine (Rodriguez-Fernandez et al., 2006a). Because many different data sets were available (data set D_1, $D_{2,A}$, $D_{2,B}$, $D_{3,A}$ and $D_{3,B}$) from the experimental work of Palatsi et al. (2012), the calibrations of the proposed models were performed in a sequential mode as explained below:

LCFA-M1 model

Step 1. Data set D_1 was used to determinate the LS-estimate $\theta_1 = [k_{ads}\ q_{sat}\ k_{des}]$ for the LCFA-adsorption model of Eq. (3.3.34). Because the experimental design was such that the adsorption process was independent from the biological process (inactivated biomass), the calibration of θ_1 was performed in batch-mode. The relative SA indices for θ_1 was obtained conditional on an uniform distribution $g(\theta_1)$ where the i-th parameter was assumed independent.

Step 2. Data set D_2 was used to estimate the initial methanogenic populations (X_{ac} and X_{h2}), the organic matter recirculation in the system (initial X_{dec}) and the first-order hydrolytic kinetics (k_{hyd}) for sludge-A and sludge-B. The parameter vector $\theta_2 = [X_{dec}\ X_{ac}\ X_{h2}\ k_{hyd}]$ was constrained over a finite range interval given by an assigned $g(\theta_2)$. Sensitivity indices were calculated. During the LS-estimation and SA for the parameter vector θ_2, parameters associated with the LCFA-inhibition process (e. g., K_{Xfa}, K_{Xac}, X_{c18} and X_{c16}) were kept constant on their arbitrary optimization-starting-point values. Because the SMA assays are LCFA-free, the parameters associated with the LCFA-inhibition process cannot influence the misfit function J. Nominal values for the remaining model parameters (see Table 3.1) were assumed according to Rosen and Jeppsson (2006).

Table 3.1: Biochemical parameter values assumed from Rosen and Jeppsson (2006).

Parameter	Units	Value	Parameter	Units	Value
$k_{m,fa}$	d^{-1}	6	Y_{fa}	kg $\mathrm{CO}D_X$·kg $\mathrm{CO}D_S^{-1}$	0.06
$K_{S,fa}$	kg COD m^{-3}	0.4	Y_{h2}	kg $\mathrm{CO}D_X$·kg $\mathrm{CO}D_S^{-1}$	0.06
$k_{m,ac}$	d^{-1}	8	Y_{ac}	kg $\mathrm{CO}D_X$·kg $\mathrm{CO}D_S^{-1}$	0.05
$K_{S,ac}$	kg COD m^{-3}	0.15	α	kg $\mathrm{CO}D_X$·kg $\mathrm{CO}D_S^{-1}$	0.3
$k_{m,h2}$	d^{-1}	35	β_{ac}	kg $\mathrm{CO}D_S$·kg $\mathrm{CO}D_S^{-1}$	0.0784
$K_{S,h2}$	kg COD m^{-3}	$7.0{\cdot}10^{-6}$	β_{h2}	kg $\mathrm{CO}D_S$·kg $\mathrm{CO}D_S^{-1}$	0.0196
$K_{I,h2,fa}$	kg COD m^{-3}	$5.0{\cdot}10^{-6}$	f_{Xi}	kg $\mathrm{CO}D_S$·kg $\mathrm{CO}D_S^{-1}$	0.3

Step 3. The SA is performed over data set $D_{3,\mathrm{A}}$ and $D_{3,\mathrm{B}}$ in order to evaluate their relative quality for the estimation of $\theta_{3,\mathrm{M1}} = [K_{Xfa}\ K_{Xac}]$ and $\theta_{4,\mathrm{M1}} = [X_{c18}\ X_{c16}]$. The high-informative data set is used for the calibration of $\theta_{3,\mathrm{M1}}$ and $\theta_{4,\mathrm{M1}}$.

Step 4. The parameter vector $\theta_{3,\mathrm{M1}}$, estimated within the high-informative data set (Step 3), is used to decrease the under-determination of the low-informative calibration scenario: the SA is run in order to assess the information-gain. The LCFA-degrader initial concentration parameter $\theta_{4,\mathrm{M1}}$ is estimated within the improved calibration scenario for the low-informative data set. The idea is that the high-informative data set is used to calibrate the model, while the low-informative data set is used to "semi-validate" the structure of the proposed model.

When one sub-model at a time SA is performed, it is possible to overlook interactions among parameters in different sub-models (type II error).

LCFA-M2 model

The same sequential calibration mode was performed for the second proposed model LCFA-M2 with the only difference that the parameter vectors $\theta_{3,\mathrm{M2}} = [K_{Xfa}\ r_{max}\ \gamma]$ and $\theta_{4,\mathrm{M2}} = [X_{c18}\ X_{c16}]$ were only calibrated for the high-informative data set obtained in Step 3.

3.4 Results and Discussion

3.4.1 Initial Parameter Estimation

The experimental design was such that data sets D_1 and D_2 were independent from the biological LCFA degradation-inhibition process (focus of the present study), or independent from any introduced ADM1 model modifications (LCFA-M1/LCFA-M2). Consequently, the calibration of θ_1 and θ_2 can be performed in a batch-mode and their values can be used in further data set modeling.

Table 3.2 summarized the LS-estimates and the sensitivity indices of the parameter θ_1 . We observe that the estimated saturation coefficient (q_{sat}) for bentonite is higher than for inactivated biomass. Thus, bentonite seems to be a better adsorbent media than inactivated biomass. Those results are in accordance with Mouneimne et al. (2004) where the authors observed a higher affinity of oleate for adsorption onto bentonite than onto biomass. The desorption rate for bentonite, $k_{des,ben}$, is higher than for inactivated biomass, $k_{des,bio}$, which suggests that the oleate adsorbed onto bentonite can be more easily desorbed than the oleate adsorbed onto biomass. Nevertheless, it must be considered that the main effect index (S_i) for the adsorption/desorption rate coefficients (i. e. k_{ads} and k_{des}) are relatively low in comparison to the total effect index (S_{Ti}). In particular, for the inactivated biomass adsorbent, it appears that there is not enough data-information in order to estimate precisely the values of k_{ads} and k_{des}. Thus, the causal interpretation of those parameters should be considered with reserve. However, since the estimated values are in accordance with other studies, we expect that the calibrated adsorption-model is a reasonable model of the process.

Table 3.3 summarized the LS-estimates of the parameter θ_2 . The relative sensitivity indices, conditional on data set $\mathrm{D}_{2,\mathrm{A}}$ and $\mathrm{D}_{2,\mathrm{B}}$, are also reported in Table 3.3. The main effect index, S_i, explains almost all the variance of the misfit function J, which implies that the parameter θ_2 can be well determined. This result was expected since the under-determination of the estimation problem was reduced by assuming the values of the ADM1's maximum uptake rates and half saturation constants of the anaerobic

Table 3.2: Sensitivity indexes and LS-estimates of the adsorption-parameter vector θ_1 for data set D_1.

Parameter	Units	Data Set	$g(\theta)$	S_i	S_{Ti}	LS-estimate
$k_{ads,ben}$	d^{-1}	D_1	*Unif* (1e-2, 2)	8.93	31.84	0.35
$k_{des,ben}$	d^{-1}	D_1	*Unif* (1e-2, 2)	9.46	25.99	0.2
$q_{sat,ben}$	kg COD kg TS^{-1}	D_1	*Unif* (1e-2, 5)	46.14	79.49	0.82
$k_{ads,bio}$	d^{-1}	D_1	*Unif* (1e-2, 2)	16.31	58.02	0.4
$k_{des,bio}$	d^{-1}	D_1	*Unif* (1e-2, 2)	8.64	37.35	0.64
$q_{sat,bio}$	kg COD kg TS^{-1}	D_1	*Unif* (1e-2, 5)	25.10	72.14	2.95

Table 3.3: Sensitivity indexes and LS-estimates of parameter vectors θ_2 for sludge-A and sludge-B.

Parameter	Units	Data Set	$g(\theta)$	S_i	S_{Ti}	LS-estimate
k_{hyd}	d^{-1}	$D_{2,A}$	*Unif* (1e-4, 2)	5.10	7.59	$2.98{\cdot}10^{-3}$
X_{ac}	kg COD m^{-3}	$D_{2,A}$	*Unif* (1e-4, 5)	30.23	33.16	0.0727
X_{h2}	kg COD m^{-3}	$D_{2,A}$	*Unif* (1e-4, 5)	30.79	33.69	$3.61{\cdot}10^{-3}$
X_{dec}	kg COD m^{-3}	$D_{2,A}$	*Unif* (1e-4, 2)	28.43	31.68	0.36
k_{hyd}	d^{-1}	$D_{2,B}$	*Unif* (1e-4, 2)	13.06	18.60	$8.19{\cdot}10^{-4}$
X_{ac}	kg COD m^{-3}	$D_{2,B}$	*Unif* (1e-4, 5)	12.81	15.89	0.02391
X_{h2}	kg COD m^{-3}	$D_{2,B}$	*Unif* (1e-4, 5)	12.08	14.90	$7.19{\cdot}10^{-4}$
X_{dec}	kg COD m^{-3}	$D_{2,B}$	*Unif* (1e-4, 2)	54.18	59.90	1.41

biomass populations involved (Rosen and Jeppsson, 2006), as reported in supporting information (Table 3.1).

Biomass concentration in SMA tests for sludge-A was slightly higher than in sludge-B (Table 3.3). Contrary, the residual slowly-biodegradable organic-matter, X_{dec}, is significantly lower for sludge-A than for sludge-B (fresh granules obtained from a running UASB reactor from a fruit juice processing industry).

3.4.2 Data Set Selection for LCFA-M1 Calibration

The relative sensitivity analysis indices (S_i and S_{Ti}) of the parameters K_{Xfa}, K_{Xac}, X_{c18} and X_{c16} were reported in Table 3.4. SA indices are conditional on their relative data sets $D_{3,A}$ and $D_{3,B}$. The main and the first-order interaction effects explained the 91.6% and the 92.5% of the total misfit-function variance for data set $D_{3,A}$ and $D_{3,B}$, respectively. The remaining variance is explained by higher-order interactions of the parameters. We observe that the main effect indices relative to the parameter $\theta_{3,M1}$ are lower for data

Table 3.4: Sensitivity indexes of parameter vectors $\theta_{3,\mathrm{M1}}$ and $\theta_{4,\mathrm{M1}}$ for sludge-A and sludge-B.

Parameter	Units	Data Set	$g(\theta)$	S_i	S_{Ti}
K_{Xfa}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4, 2)	38.4	53.8
K_{Xac}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4, 2)	1.4	9.2
X_{c18}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4, 5)	18.5	29.4
X_{c16}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4, 5)	24.6	35.1
K_{Xfa}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{B}}$	*Unif* (1e-4, 2)	35.4	71.8
K_{Xac}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{B}}$	*Unif* (1e-4, 2)	0.2	2.8
X_{c18}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{B}}$	*Unif* (1e-4, 5)	11.82	7.9
X_{c16}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{B}}$	*Unif* (1e-4, 5)	15.0	43.2

set $\mathrm{D}_{3,\mathrm{B}}$ than for data set $\mathrm{D}_{3,\mathrm{A}}$. Thus, the high-informative data set of our case study is data set $\mathrm{D}_{3,\mathrm{A}}$. The parameter $\theta_{3,\mathrm{M1}}$ estimated within data set $\mathrm{D}_{3,\mathrm{B}}$ can be considered unidentifiable but still important in order to correctly fit data since its total effect index is not negligible.

We observe that for almost all the parameters the difference between S_i and S_{Ti} is consistently higher for data set $\mathrm{D}_{3,\mathrm{B}}$ then for data set $\mathrm{D}_{3,\mathrm{A}}$, which implies that for data set $\mathrm{D}_{3,\mathrm{B}}$ the interaction effects between parameters are stronger then for data set $\mathrm{D}_{3,\mathrm{A}}$. In particular, for data set $\mathrm{D}_{3,\mathrm{B}}$, the first-order interaction of K_{Xfa} with X_{c18} and X_{c16} explains the 28.6% of the total variance of the misfit- function, while for data set $\mathrm{D}_{3,\mathrm{A}}$ this interaction effect accounts only for a 6.4% of the total variance of J (results not shown). Moreover, the main effect indices S_i relative to $\theta_{4,\mathrm{M1}}$ are higher for data set $\mathrm{D}_{3,\mathrm{A}}$ then for data set $\mathrm{D}_{3,\mathrm{B}}$ (see Table 3.4).

We conclude that data set $\mathrm{D}_{3,\mathrm{A}}$ is more appropriate than data set $\mathrm{D}_{3,\mathrm{B}}$ in order to estimate the parameters associated with the LCFA-inhibition process K_{Xfa} and K_{Xac}. data set $\mathrm{D}_{3,\mathrm{A}}$ is considered as the high-informative data set, while data set $\mathrm{D}_{3,\mathrm{B}}$ is considered as the low-informative in relation to our model calibration set-up.

From the above SA-results, it was decided to use the experimental design of sludge-A in order to estimate the parameter $\theta_{3,\mathrm{M1}}$ for the LCFA-M1 model. Here, we limit our discussion on the model simulation outcomes and on the goodness of the fit, since data set $\mathrm{D}_{3,\mathrm{A}}$ (and $\mathrm{D}_{3,\mathrm{B}}$) was presented and discussed in detail in Palatsi et al. (2012).

3.4.3 LCFA-M1 Model Calibration. Sludge-A

Figure 3.4.1 shows the simulation of the liquid-solid LCFA phases, the Ac concentration and the CH_4 production, for the vials with bentonite addition T_A (dash line) and the control vial C_A (continuous line). The goodness of the fit is quantified within the root-mean, squared-error (rmse) statistic. Simulation results of the batch experiments with an increasing oleate concentration (also included in data set $\mathrm{D}_{3,\mathrm{A}}$) are reported in Figure 3.4.2. The LS-estimated parameters are summarized in Table 3.5.

We observe from Figure 3.4.1, that the oleate concentration in the liquid ($\mathrm{C18}_\mathrm{l}$) is well described by the adsorption model (i. e. $S_{c18,l}$ and $S_{c18,l}$ model outcomes). The adsorption process is a very fast process if compared with the biological-mediated process, as reported by Hwu et al. (1998). The sampling frequency of the measurements was insufficient to ketch the fast-adsorption dynamics at the beginning of the experiment. The uncertainty of the adsorption-parameter vector θ_1 can influence only slightly the misfit function value. Remark that the estimation of the parameter vector θ_2 can be achieved by high precision. Thus, we expect that the SA performed over parameter vectors θ_3 and θ_4 should lead to a negligible type II error, i. e. assessing as non-important an important parameter.

We observe from Figure 3.4.1 an accumulation of oleate on the solid phase ($\mathrm{C18}_\mathrm{s}$), which degradation is followed by a palmitate accumulation in the solid phase ($\mathrm{C16}_\mathrm{s}$). Pereira et al. (2002) also identified palmitate as key intermediate specie during oleate degradation in not-adapted systems. The simulated palmitate concentration of LCFA-M1 were almost entirely adsorbed into biomass ($S_{c16,s} \approx S_{c16,ben}$ since $S_{c16,\mathrm{ben}} \approx 0$), confirming the microscopy observations of granules performed on day 10 (Palatsi et al., 2012). According to the model simulation of LCFA-M1 for the strategy T_A and the control C_A, we observed that the $\mathrm{C16}_\mathrm{bio}$ concentration time evolution was approximately the same. This evidenced that the strategy tested in T_A vials (bentonite addition after LCFA pulse) is not efficient for LCFA-inhibition prevention.

The main problem with the LCFA-M1 model is the poor data-fit of the accumulation process of $\mathrm{C16}_\mathrm{s}$ (see Figure 3.4.1). In fact, the modeled degradation of $\mathrm{C16}_\mathrm{s}$ is delayed almost 10 days (i. e. approximately from day 25 to day 35) if compared with data. Remark that the misfit function favors the fit of the CH_4 measurements. Consequently, in order to fit well the CH_4 measurements, the LCFA-M1 model artificially extends the LCFA-inhibition effect with a larger $\mathrm{C16}_\mathrm{bio}$ accumulation. The problem is that the inhibition function for the acetoclastic population I_{Xac} (Eq. 3.3.36) depends directly on the LCFA_s concentration present in the system. This delay between the LCFA-consumption and the methane-formation was also reported in previous experiments (Palatsi et al., 2009). Despite the model-structure limitation of LCFA-M1, it is capable to reproduce

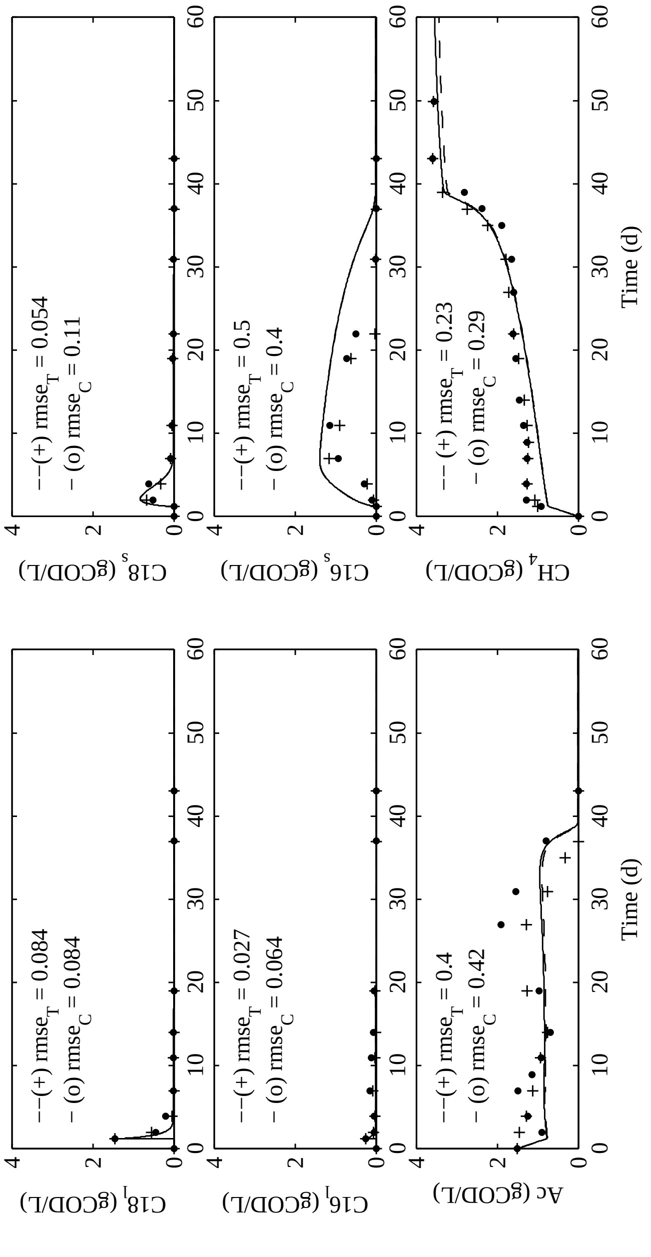


Figure 3.4.1: Calibration of the LCFA-M1 model with Sludge-A (data set $D_{3,A}$). The bentonite addition (T_A) model-outcome (dash line) and observations (cross dots) are compared with the control-experiment (C_A) model-outcome (continuous line) and observations (circle dots).

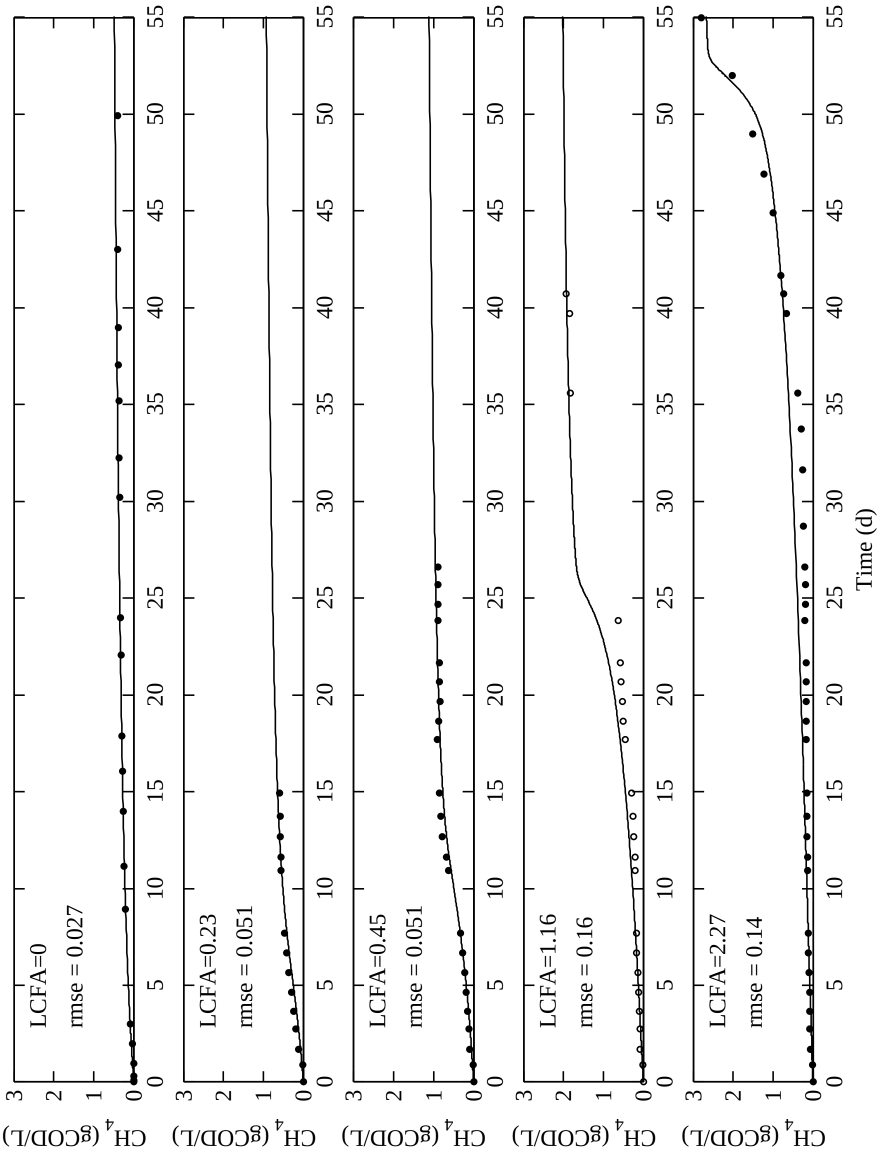


Figure 3.4.2: LCFA-M1 model-fit for sludge-A LCFA-toxicity assay. The empty-circle measurments were not used for model calibration.

Table 3.5: LS-estimate of parameter vectors $\theta_{3,\mathrm{M1}}$ and $\theta_{4,\mathrm{M1}}$ for sludge-A.

Parameter	Units	Data Set	LS-estimate
K_{Xfa}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	0.324
K_{Xac}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	0.045
X_{c18}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	0.496
X_{c16}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	0.020

reasonably well the main trends of the system.

The LS-estimate for the LCFA-inhibition parameter of acetogenic-degraders, K_{Xfa}, was 0.324 kgCODm^{-3} (see Table 3.5), while the LCFA-inhibition parameter of acetoclasitic-degraders, K_{Xac}, was 0.045 kgCODm^{-3}. This result suggests that the acetoclastic population is more sensitive to the LCFA-inhibition than it is the acetogenic population, in accordance with previous reports (Salminen et al., 2000; Lokshina et al., 2003; Palatsi et al., 2010). According to the obtained model parameters, the initial acetogenic-degraders structure was dominated by the oleate-degraders X_{c18}, creating a potential condition for a palmitate-accumulation that may lead to a long lasting LCFA inhibition of the system.

3.4.4 LCFA-M1 Model Structure Semi-Validation. Sludge-B

If the two data sets $\mathrm{D}_{3,\mathrm{A}}$ and $\mathrm{D}_{3,\mathrm{B}}$ would be obtained within the same sludge then model validation would be possible. Since it is not the case, the LS-estimates of X_{c18} and X_{c16} for sludge-A cannot be used to validate the model over data of sludge-B. However, we will use the improper name of “semi-validation” to refer to the scenario where we calibrate $\theta_{4,\mathrm{M1}}$ for sludge-B conditional on the parameter vector $\theta_{3,\mathrm{M1}}$ that was calibrated for sludge-A (data set $\mathrm{D}_{3,\mathrm{A}}$). In this way, we make the strong assumption that for sludges of different origins the LCFA-inhibition effect depends only on the LCFA-population structure distribution, while the LCFA-resiliance (K_{Xfa} and K_{Xac}) of the biomass is approximately constant.

Apart of performing the semi-validation of LCFA-M1, the lack of information of data set $\mathrm{D}_{3,\mathrm{B}}$ (sludge-B) can be improved when perfect knowledge is assumed over the parameter $\boldsymbol{\theta_{3,\mathrm{M1}}}$. Because the sensitivity indices relative to $\theta_{3,\mathrm{M1}}$ would be zero for this perfect-knowledge, SA-scenario, a small amount of uncertainty was added to $\theta_{3,\mathrm{M1}}$. The uncertainty of $\theta_{3,\mathrm{M1}}$ was modeled within a normal distribution. Table 3.6 summarized the repeated SA for sludge-B observations. We observe that the variance of the misfit function is mainly explained within the initial concentration of the LCFA-degraders (X_{c18} and X_{c16}); the S_i index improves (see Table 3.6), while the K_{Xfa} interaction first-

Table 3.6: Sensitivity indexes and LS-estimates of parameter vectors $\theta_{3,\mathrm{M1}}$ and $\theta_{4,\mathrm{M1}}$ for sludge-B. An informative SA scenario is considered where the parameter vector $\theta_{3,\mathrm{M1}}$ is known with a low degree of uncertainty modeled with a normal distribution $g(\theta_{3,\mathrm{M1}})$.

Parameter	Units	Data Set	$g(\theta)$	S_i	S_{Ti}	LS-estimate
K_{Xfa}	kg COD m^{-3}	$D_{3,B}$	*Norm* (0.324, 0.023^2)	1.1	7.9	0.324
K_{Xac}	kg COD m^{-3}	$D_{3,B}$	*Norm* (0.045, 0.006^2)	0.0	0.0	0.045
X_{c18}	kg COD m^{-3}	$D_{3,B}$	*Unif* (1e-4, 5)	48.6	67.7	0.067
X_{c16}	kg COD m^{-3}	$D_{3,B}$	*Unif* (1e-4, 5)	29.1	49.5	0.242

order effect with X_{c18} and X_{c16} decreases to only 2.9% (not shown in Table 3.6). If we compare in Table 3.4 and 3.6 the values of S_i relative to data set $D_{3,B}$, we can observe that the estimation-precision of parameters X_{c18} and X_{c16} is improved when the a-priori information about K_{Xfa} and K_{Xac} rules out unrealistic possibilities. The model-fit to data of LCFA-M1 for the semi-validation scenario is represented in Figure 3.4.3 (LCFA-batch assays at increasing oleate concentrations model-fit results are represented in Figure 3.4.4).

Similarly as for sludge-A, the adsorption model cannot be evaluated because of the low sampling frequency; the low rmse value for $C18_l$ and $C16_l$ should be considered with reserve. The simulated LCFA$_s$, $S_{fa,s}$, was equivalent to the LCFA-bentonite adsorbed concentration, $S_{fa,ben}$, because bentonite was mixed with the LCFA-inhibition concentration before its addition to the anaerobic system (Palatsi et al., 2012). If the control experiment (C_B) is considered, the simulation reproduce quite well the $C18_s$ and $C16_s$ observations, while if we consider the prevention-strategy experiment (T_B) the model under-estimates those data. Moreover, we observe a relevant misfit for Ac data if the prevention-strategy experiment (T_B) is considered: the Ac accumulation reproduced within the LCFA-M1 model is not detected by the measurements. The misfit of $C18_s$ and Ac are necessary in order to correctly reproduce the methane production measurements. Since the methane measurements are of high-fidelity (more available data, including batch assays with increasing oleate concentrations, D_3) we can reasonably suspect that the experimental results of $C18_s$ and Ac at day 7 were erroneous (from COD balance). Note that $C18_s$ and Ac measurements were conducted within a vial sacrifice (Palatsi et al., 2012) and thus it is possible that the day-7 vial was an outlier.

Considering sludge-B informative scenario (see Table 3.6), the LS-estimates for the initial concentration of X_{c18} and X_{c16} were 0.067 kgCODm^{-3} and 0.242 kgCODm^{-3}, respectively. The high palmitate-degraders population concentration can be explained

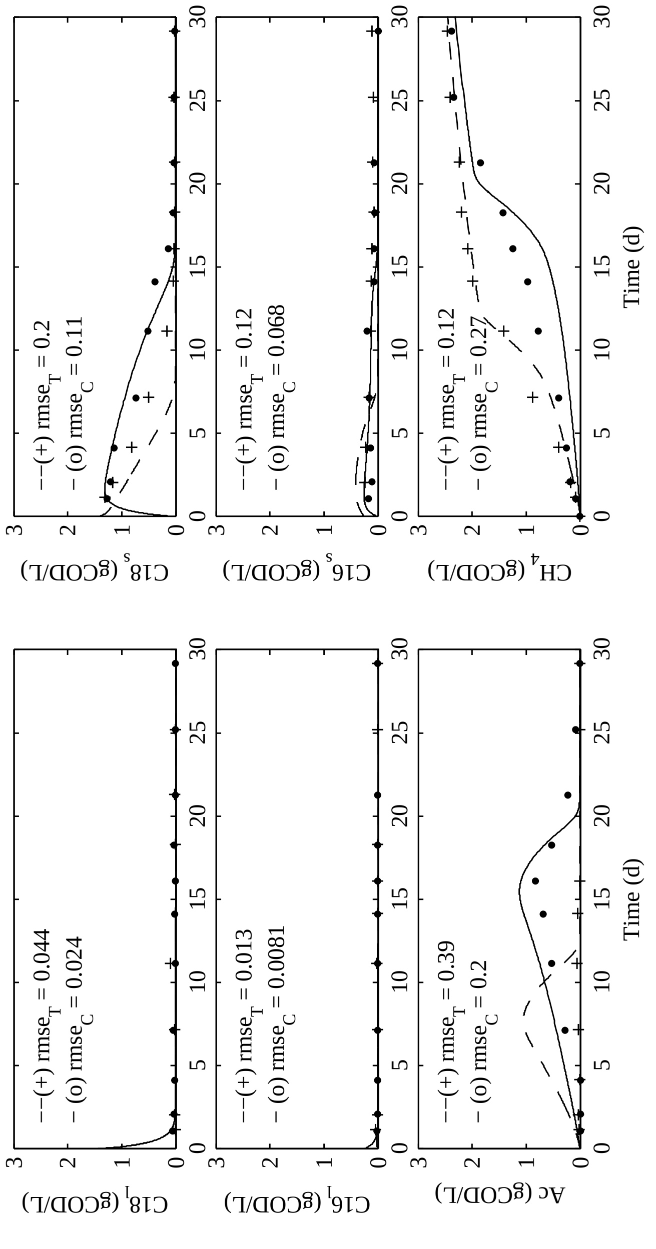


Figure 3.4.3: Semi-validation of the LCFA-M1 model with Sludge-B (data set $D_{3,B}$). The bentonite addition (T_B) model-outcome (dash line) and observations (cross dots) are compared with the control-experiment (C_B) model-outcome (continuous line) and observations (circle dots).

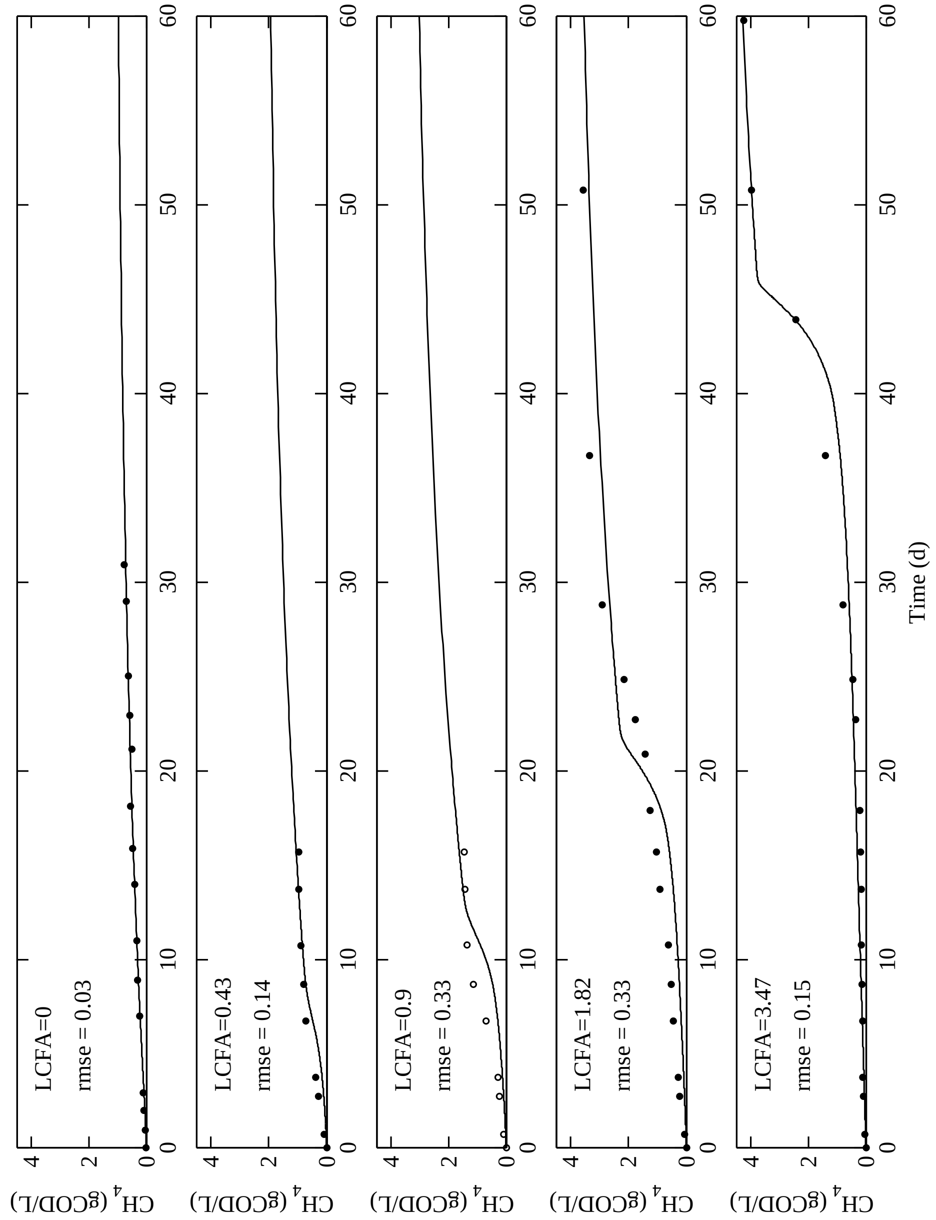


Figure 3.4.4: LCFA-M1 model-fit for sludge-B LCFA-toxicity assay. The empty-circle measurments were not used for model calibration.

in part for the absence of the palmitate accumulation as observed for sludge-A. During molecular profiling of biomass A and B, by PCR-DGGE techniques (Palatsi et al., 2012) it was not possible to confirm this hypothesis. Now, results of process modeling give new insights about the importance of the specific microbial structure of *ß*-oxidative organisms. Remark that the estimated X_{dec} concentration for sludge-B was higher than the X_{dec} for sludge-A (during θ_2 estimation, Table 3.3). This fact was previously pointed out by Pereira et al. (2004) and Palatsi et al. (2010) as a possible factor influencing the LCFA-degradation dynamics, since the presence of other biodegradable substrates (considered in X_{dec} pull) may enhance LCFA-degradation rates (Kuang et al., 2006).

The LCFA-M1 model is able to reproduce well the main system trends also for sludge-B, confirming the adsorptive nature of LCFA inhibitory process, with the simulated differences between T_B and C_B vials (Figure 3.4.3). The results confirm the opportunity of using bentonite as a synthetic adsorbent (additive) to interfere in the LCFA-adsorption-inhibition process (Palatsi et al., 2012). Moreover, under a slight LCFA-inhibition of the system, the LCFA-M1 model seems to confirm the hypothesis that the acetogenic and the acetoclastic LCFA-inhibition coefficients are invariant within different sludges. However, in order to predict the evolution of an anaerobic system the relative LCFA-degraders population structure distribution should be known or estimated.

3.4.5 LCFA-M2 Model Calibration. Sludge-A

The LCFA-M2 model SA is resumed in Table 3.7, where the main and the total indices are reported for the respective model-parameters. Note that the parameter $\theta_{4,\mathrm{M2}} = [X_{c18}\ X_{c16}]$ alone explains almost all the variance of the misfit function J (i. e. 87%). This implies that we can estimate $\theta_{4,\mathrm{M2}}$ with high precision within the LCFA-M2 model structure and data set $\mathrm{D}_{3,\mathrm{A}}$. On the other hand, the parameter vector $\theta_{3,\mathrm{M2}} = [K_{Xfa}\ r_{max}\ \gamma]$ affects J only within interactions and, thus, it cannot be well determined. However, the estimation of the parameter vector $\theta_{3,\mathbf{M2}}$ is still very important in order to fit well the collected data. Moreover, we report that the first-order interaction index $S_{i,j}$ (not shown in Table 3.7) for all the parameter-pairs was negligible. Thus, the presence of higher-order interactions suggests that the interaction structure is quite complex.

The LS-estimate of parameter vectors $\theta_{3,\mathrm{M2}}$ and $\theta_{4,\mathrm{M2}}$ for sludge-A are reported in Table 3.4. The LS-estimate of the parameter K_{Xfa} was 0.260 kgCODm^{-3}. This value was of the same order of magnitude of the LS-estimate relative to the LCFA-M1 model (Table 3.3). The LS-estimate for the parameter γ was higher than one (γ = 2.41); for example, when the average damage of the cell-functionality D_{Xac} is of 25% then we can expect that only the 50% (=(1-0.25)$^{2.41}$) of the Ac-degraders have fully re-activated

Table 3.7: Sensitivity indexes and LS-estimates of parameter vectors $\theta_{3,\mathrm{M2}}$ and $\theta_{4,\mathrm{M2}}$ for sludge-A.

Parameter	Units	Data Set	$g(\theta)$	S_i	S_{Ti}	LS-estimate
K_{Xfa}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4, 2)	1.14	4.92	0.260
r_{max}	d^{-1}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4 2)	3.07	8.98	0.066
γ	-	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1 5)	0.54	4.91	2.415
X_{c18}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4, 5)	41.16	45.51	0.300
X_{c16}	kg COD m^{-3}	$\mathrm{D}_{3,\mathrm{A}}$	*Unif* (1e-4, 5)	45.91	50.74	0.053

their methanogenic pathways.

The initial concentrations of LCFA-degraders estimated within the LCFA-M2 model (see Table 3.7) are qualitatively the same as those of the LCFA-M1 model (see Table 3.4). The oleate-degraders are found to be the dominant population in the sludge-A, explaining the higher or longer palmitate-accumulation respect to sludge-B. The model-fit of data set $\mathrm{D}_{3,\mathrm{A}}$ within the LCFA-M2 model is reported in Figure 3.4.5. Simulation results of the batch experiments with an increasing LCFA concentration (also included in data set $\mathrm{D}_{3,\mathrm{A}}$) are reported in Figure 3.4.6. We observe from Figure 3.4.5 that the misfit of $\mathrm{C18_l}$ is practically the same as for the LCFA-M1 (see Figure 3.4.1) since the adsorption-model is the same and the adsorption process is very fast when compared with the biological processes. If we compare the model-fit of LCFA-M2 and LCFA-M1 then the LCFA-M2 model gives as a slightly worse result for the oleate concentration on the solid phase ($S_{c18,s} \approx S_{c18,ben}$ since $S_{c18,ben} \approx 0$). However, the LCFA-M2 model performs very well if $\mathrm{C16_s}$ is considered. Note that the LCFA over-accumulation artifact observed when the LCFA-M1 model is used to simulate the sludge-A experiment (see Figure 3.4.1) is not present when the LCFA-M2 model is considered (see Figure 3.4.5). In fact, we observe from Figure 3.4.5 that the period that goes from the total $\mathrm{C16_s}$ depletion to the re-start of the $\mathrm{CH_4}$-production (delay of ten days) is correctly simulated within the LCFA-M2 model. This is because the LCFA-inhibition effect in the LCFA-M2 model is not directly dipendent on the current value of the concentration of the LCFA-adsorbed on biomass. Thus, no artificial delay of LCFA-concentration is necessary in order to correctly fit the methane production measurements. In our case, the majority of the delay period is characterized by an increase of the healthy-state H_{Xac}. The only active bacteria are the acetogens that promote the Ac-accumulation. The simulated Ac-accumulation is quite pronounced in order to satisfy the COD balance (see Figure 3.4.5). In particular, at the start-up of the experiment, the simulated degradation of the acetate-pulse is faster than the Ac-measurement seems to suggest. However, the

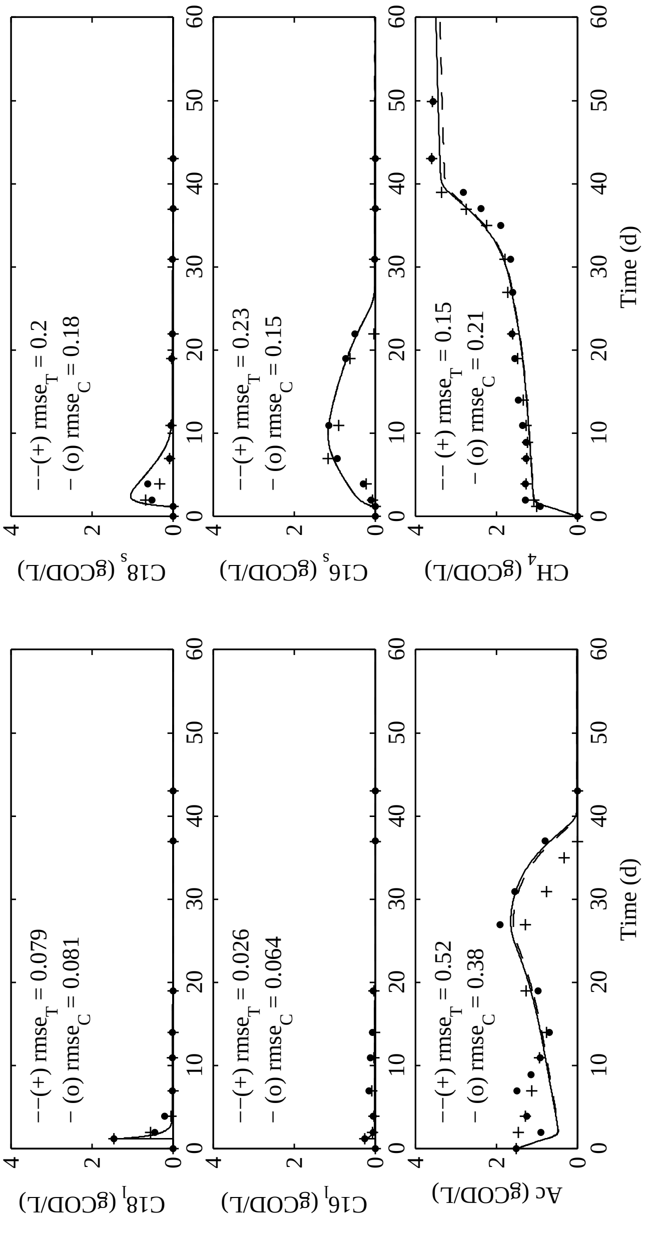


Figure 3.4.5: Calibration of the LCFA-M2 model with Sludge-A (data set $D_{3,A}$). The bentonite addition (T_A) model-outcome (dash line) and observations (cross dots) are compared with the control-experiment (C_A) model-outcome (continuous line) and observations (circle dots).

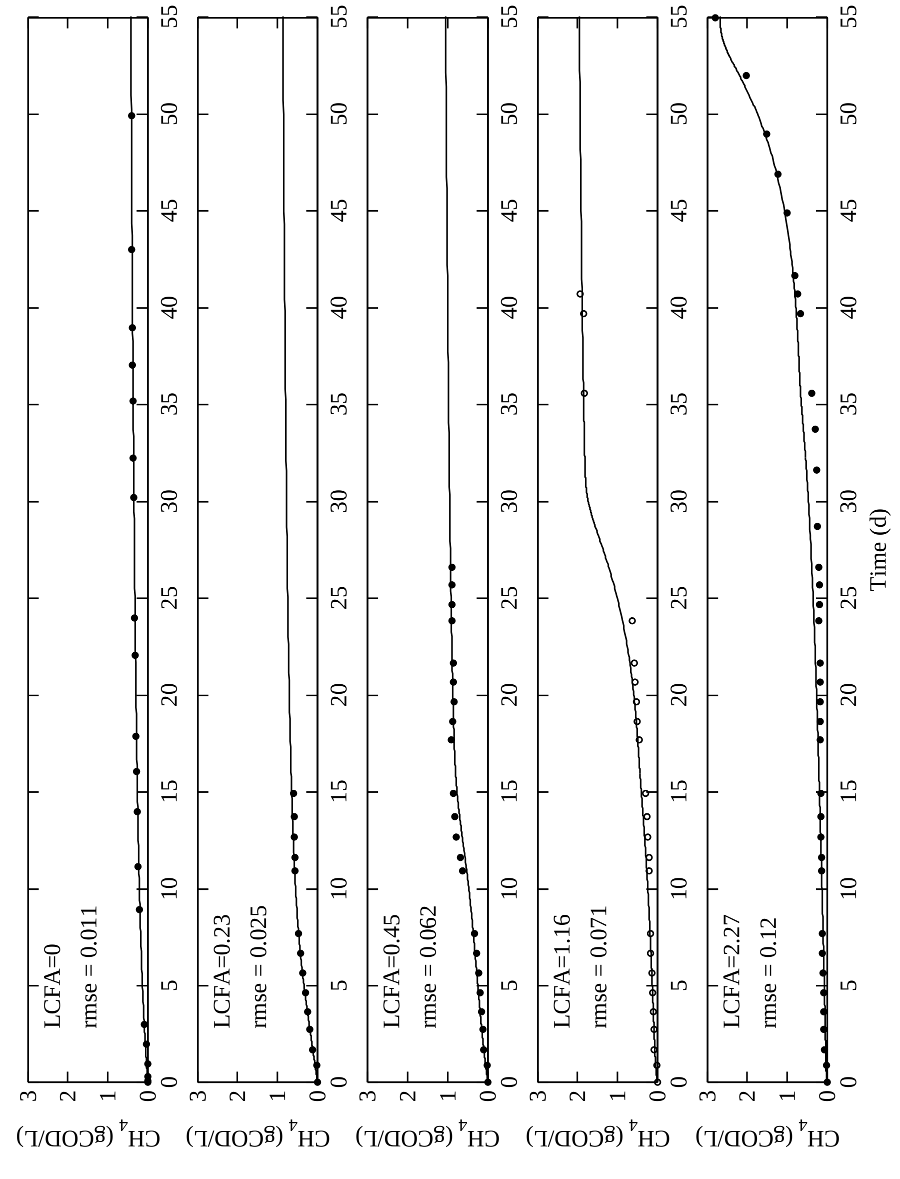


Figure 3.4.6: LCFA-M2 model-fit for sludge-A LCFA-toxicity assay. The empty-circle measurments were not used for model calibration.

start-up CH_4-production data is very well fitted. Because we are more confident in the CH_4 measurements (more data available with a low measurement error), the LCFA-M2 model-simulation probably evidence a problem with the few-first Ac-samples. Note that the LCFA-M1 model was not able to represent correctly the CH_4-production at the start-up of the experiment (see Figure 3.4.1).

The LCFA-M2 model was superior to the LCFA-M1 model in order to describe the CH_4-production data from the toxicity assays (compare also Figure 3.4.2 and 3.4.6 in supporting information) and the LCFA-monitored observations (compare Figure 3.4.1 and 3.4.5). However, the LCFA-M2 model use one degree of freedom (parameter) more then the LCFA-M1 model and thus it is expected to be more flexible for data fitting. Since the LCFA-inhibition parameters K_{Xfa}, r_{max} and γ are difficult to identify with data set $D_{3,A}$, the semi-validation of LCFA-M2 with data set $D_{3,B}$ would probably produced unsatisfactory results.

Provided that the LCFA-M2 model is calibrated only on data from the experimental set-up of sludge-A, its use (e. g., optimization and control routines) should be constrained only over its calibration domain. Extrapolation with the LCFA-M2 model (calibrated over sludge-A) should be avoided because of its over-parametrized structure. Therefore, if extrapolation is considered the LCFA-M1 model seems to be more robust then the LCFA-M2 model. However, the LCFA-M2 model performes better on specific interpolation rotines since its structure can describe the LCFA-inhibition process on a lower-scale than the LCFA-M1 model structure.

3.5 Conclusions

Two new LCFA-inhibition models (i. e. LCFA-M1 and LCFA-M2) were proposed that can be easily integrated into the full ADM1 framework. The adsorptive nature of LCFA over granular biomass and specific LCFA-degrader populations were included in both models. The main distinction between the two models was on how the acetoclastic LCFA-inhibitory phenomena was represented: i) a common non-competitive inhibition function (LCFA-M1) or ii) a new state variable that accounts directly for the damage of the cell-functionality (LCFA-M2). Both models were proven to reproduce the main trends of a LCFA-inhibited system operated in a wide range of experimental designs. However, the simpler LCFA-M1 model was not able to reproduce correctly the dynamics of the the LCFA-degradation as the LCFA-M2 model did. Causal modeling within the two proposed models confirmed that the acetoclasitic population is more sensible to the LCFA-inhibition than it is the acetogenic population. In addition, it was evidenced that the distribution of the saturated/unsaturated degraders plays an important role on the system evolution.

Acknowledgements

This work was funded by the Spanish Ministry of Science and Innovation (project CTM 2010-18212).

Bibliography

Alves, M. M., Picavet, M. A., Pereira, M. A., Cavaleiro, A. J., Sousa, D. Z., 2007. Novel anaerobic reactor for the removal of long chain fatty acids from fat containing wastewater.

Angelidaki, I., Ellegaard, L., Ahring, B. K., 1999. A comprehensive model of anaerobic bioconversion of complex substrates to biogas. Biotechnology and Bioengineering 63 (3), 363–372.

Bastone, D. J., Keller, J., Angelidaki, I., Kalyuzhnyi, S. V., Pavlostathis, S. G., Rozzi, A., Sanders, W. T. M., Siegrist, H., Vavilin, V. A., 2002. Anaerobic digestion model no.1 (ADM1). Tech. rep., IWA Publishing , UK.

Beccari, M., Majone, M., Riccardi, C., Savarese, F., Torrisi, L., 1999. Integrated treatment of olive oil mill effluents: Effect of chemical and physical pretreatment on anaerobic treatability. Water Science and Technology 40 (1), 347–355.

Desbois, A. P., Smith, V. J., 2010. Antibacterial free fatty acids: activities, mechanisms of action and biotechnological potential. Applied Microbiology and Biotechnology 85 (6), 1629–1642.

Hanaki, K., Matsuo, T., Nagase, M., 1981. Mechanisms of inhibition caused by long chain fatty acids in anaerobic digestion process. Biotechnology and Bioengineering 23(7), 1591–1610.

Hatamoto, M., Imachi, H., Yashiro, Y., Ohashi, A., Harada, H., 2007. Diversity of anaerobic microorganisms involved in long-chain fatty acids degradation in methanogenic sludges as revealed by RNA-based stable isotope probing. Applied and Environmental Microbiology 73 (13), 4119–127.

Hwu, S. H., Tseng, S. K., Yuan, C. Y., Kulik, Z., Lettinga, G., 1998. Biosorption of long-chain fatty acids in UASB treatment process. Water Research 32 (5), 1571–1579.

Kim, B. H., Gadd, G. M., 2008. Bacterial physiology and metabolism. Cambridge University Press. New York, USA.

Kuang, Y., Pullammanappallil, P., Lepesteur, M., Ho, G.-E., 2006. Recovery of oleate-inhibited anaerobic digestion by addition of simple substrates. Journal of Chemical Technology and Biotechnology 81 (6), 1057–1063.

Lalman, J. A., Bagley, D. M., 2001. Anaerobic degradation and methanogenic inhibitory effects of oleic and stearic acids. Waster Research 35 (12), 2975–2983.

Lokshina, L. Y., Vavilin, V. A., Salminen, E., Rintala, J., 2003. Modeling of anaerobic degradation of solid slaughterhouse waste. inhibition offect of long-chain fatty acids or ammonia. Applied Biochemistry and Biotechnology 109 (1-3), 15–32.

Mouneimne, A. H., Carrère, H., Bernet, J. P., Delgenès, J. P., 2004. Effect of the addition of bentonite on the anaerobic biodegradability of solid fatty wastes. Environmental Technology 25 (4), 459–469.

Nielsen, H. B., Ahring, B. K., 2006. Responses of the biogas process to pulses of oleate in reactors treating mixtures of cattle and pig manure. Biotechnology and Bioengineering 95 (1), 96–105.

Oakley, J. E., O'Hagan, A., 2004. Probabilistic sensitivity analysis of complex models: a Bayesian approach. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66 (3), 751–769.

Palatsi, J., Affes, R., Fernandeza, B., Pereira, M. A., Alves, M. M., Flotats, X., 2012. Influence of adsorption and anaerobic granular sludge characteristics on long chain fatty acids inhibition process. Water Research - (x), –, In Press.

Palatsi, J., Illa, J., Prenafeta-Boldú, F. X., Laureni, M., Fernandez, B., Angelidaki, K., Flotats, X., 2010. Long-chain fatty acids inhibition and adaptation process in anaerobic thermophilic digestion: batch tests, microbial community structure and mathematical modelling. Bioresource Technology 101 (7), 2243–2251.

Palatsi, J., Laureni, M., Andrés, M. V., Flotats, X., Nielsen, H. B., Angelidaki, I., 2009. Recovery strategies from long-chain fatty acids inhibition in anaerobic thermophilic digestion of manure. Bioresource Technology 100 (20), 4588–4596.

Pereira, M. A., Pires, O. C., Mota, M., Alves, M. M., 2002. Anaerobic degradation of oleic acid by suspended sludge: identification of palmitic acid as a key intermediate. Water Science and Technology 45 (10), 139–144.

Pereira, M. A., Pires, O. C., Mota, M., Alves, M. M., 2005. Anaerobic biodegradation of oleic and palmitic acids: evidence of mass transfer limitations caused by long chain fatty acid accumulation onto the anaerobic sludge. Biotechnology and Bioengineering 92 (1), 15–23.

Pereira, M. A., Roest, K., Stams, A. J. M., Akkermans, A. D. L., Amaral, A. L., Pons, M.-N., Ferreira, E. C., Mota, M., Alve, M. M., 2003. Image analysis, methanogenic activity measurements, and molecular biological techniques to monitor granula sludge from an EGSB reactor fed with oleic acid. Water Science and Technology 47 (5), 181–188.

Pereira, M. A., Sousa, D. Z., Mota, M., Alves, M. M., 2004. Mineralization of LCFA associated with anaerobic sludge: Kinetics, enhancement of methanogenic activity, and effect of VFA. Biotechnology and Bioengineering 88 (4), 502–511.

Picioreanu, C., Batstone, D. J., van Loosdrecht, M. C. M., 2005. Multidimensional modelling of anaerobic granules. Water Science and Technology 52 (1-2), 501–507.

Ratto, M., Tarantola, S., Saltelli, A., 2001. Sensitivity analysis in model calibration: GSA-GLUE approach. Computer Physics Communications 136 (3), 212–224.

Rodriguez-Fernandez, M., Egea, J. A., Banga, J. R., 2006a. Novel metaheuristic for parameter estimation in nonlinear dynamic biological systems. BMC Bioinformatics 7, 483.

Rosen, C., Jeppsson, U., 2006. Aspects on ADM1 implementation within the BSM2 framework. Tech. rep., Department of Industrial Electrical Engineering and Automation, Lund University, Sweden.

Salminen, E., Rintala, J., Lokshina, L. Y., Vavilin, V. A., 2000. Anaerobic batch degradation of solid poultry slaughterhouse waste. Water Science and Technology 41 (3), 33–41.

Saltelli, A., Annoni, P., Azzini, I., Campolongo, F., Ratto, M., Tarantola, S., 2010. Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index. Computer Physics Communications 181 (2), 259–270.

Sobol', I. M., 1976. Uniformly distributed sequences with an addition uniform property. USSR Computational Mathematics and Mathematical Physics 16, 236–242.

Sousa, D. Z., Pereira, M. A., Alves, J. I., Smidt, H., Stams, A. J. M., 2008. Anaerobic microbial LCFA degradation in bioreactors. Water Science and Technology 57 (3), 439–444.

Sousa, D. Z., Pereira, M. A., Stams, A. J. M., Alves, M. M., 2007. Microbial communities involved in anaerobic degradation of unsaturated or saturated long chain fatty acids (LCFA). Applied Environmental Microbiology 73 (4), 1054–1064.

Valentine, D. L., 2007. Adaptations to energy stress dictate the ecology and evolution of the archaea. Nature Reviews Microbiology 5, 316–323.

4 Bayesian and Frequentist Inference under Comparison

Title: Estimation of parameter uncertainty for an activated sludge model using Bayesian inference: A comparison with the frequentist method (in revision at Enviromental Modelling and Software)

Živko J. Zonta[b,*], Xavier Flotats[a,b]and Albert Magrí[a]

[a]GIRO Technological Centre. Rambla Pompeu Fabra 1, 08100 Mollet del Vallès, Barcelona, Spain
[b]Department of Agrifood Engineering and Biotechnology. Universitat Politècnica de Catalunya. Campus del Baix Llobregat, Edifici D4, Esteve Terradas 8, 08860 Castelldefels, Barcelona, Spain
[*]Corresponding author

4.1 Abstract

The confidence region computed with the Fisher (FIM) or the Hessian matrix is based on the linear approximation of the model-parameter uncertainty. This linearity assumption is commonly accepted when assessing the model-parameter uncertainty for wastewater treatment activated sludge models (ASMs). The aim of this work is to test the validity of the linear-approximation assumption for an ASM-type model (non-linear in parameters) considering simultaneous storage and growth processes for the biomass. Practical identifiability was addressed exclusively considering respirometric profiles based on the oxygen uptake rate (OUR) and with the aid of a global probabilistic sensitivity analysis. Model-parameter uncertainty was thereafter estimated according to classical frequentist (linear) and Bayesian (non-linear) inferential procedures. Results were compared to evidence the strengths and weaknesses of both approaches. Since it was demonstrated that the Bayesian inference can be reduced to a frequentist approach under particular hypotheses, the first can be considered as more generalist and flexible methodology. Hence, its use is encouraged for tackling inferential issues in ASM environments.

4.2 Introduction

Respirometry is a fast and relatively inexpensive technique for wastewater activated sludge characterization based on the measurement of oxygen uptake rate (OUR) and it is commonly used for the calibration of kinetic models (Vanrolleghem et al., 1999; Magrí and Flotats, 2008). However, because respirometry data used for the calibration of wastewater models are often sparse in relation to model-complexity it is not always possible to practically identify all the parameters included (Dochain and Vanrolleghem, 2001). In practice, given the evidence provided by data, the onset of non-identifiability is gradual (Renard et al., 2010) because there are model-parameters (i) informative only for a model regime but data do not force the model in this regime, and/or (ii) informative in groups that cannot be resolved into individual components (i. e. correlated parameters). On this regard, more a parameter is practically identiable, more accurate is its estimate.

It is quite common for wastewater activated sludge models (ASMs) to assess the model-parameter accuracy (or uncertainty) within a frequentist confidence region (Marsili-Libelli and Tabani, 2002; Checchi and Marsili-Libelli, 2005; Sin et al., 2005a; Hoque et al., 2009). Briefly, the frequentist procedure is random while a model-parameter is assumed as a fixed, unknown quantity. Once the optimal parameter-value is estimated, its uncertainty is assessed from a quadratic expansion of the goodness-of-fit function

(i. e. through the Hessian) or from a local linearization in parameter of the model-outcome (i. e. involving the Fisher Information Matrix, FIM).

While "frequentists" understand model-parameters as single unknown values, "Bayesians" treat them as random variables defined by a probability distribution. In the Bayesian framework, the initial degree of belief over the parameter-uncertainty is expressed by a prior distribution (before data is collected), while the likelihood distribution indicates how likely it would be to observe the data, given a particular parameter-value. The updated degree of belief over the model-parameter is represented with a posterior distribution, which is an aggregation of the information contained in the prior and the likelihood. If the prior is non-informative (i. e. flat and uniform distribution) and the model-structure is fixed, then the relative posterior depends only on available data through the likelihood; in this case, the posterior-mode is known as the maximum likelihood estimator (MLE).

When model is linear in parameters or data sample size is large, frequentist and Bayesian procedures yield the same result (D'Agostini, 2003). From a Bayesian perspective this justifies the common scientific practice of interpreting frequentist inference (point estimates and regions) in a Bayesian fashion (i. e. probability statements about confidence regions). However, for non-linear environmental models (including ASM-type models) with sparse data the frequentist approach may under-estimate parameter-uncertainty (Omlin and Reichert, 1999; Vrugt and Bouten, 2002). Moreover, if the Hessian or the FIM matrixes are badly conditioned (i. e. high ratio-value between the highest and lowest eigenvalue of the matrix) then the advice given in most textbooks is to re-think the model and re-run the analysis or, in some cases, to get more data. Rather than re-thinking the model or collecting more data, it is common to fix the problematic, non-identifiable parameters at some nominal values and thereafter re-run the uncertainty estimation. Contrary, Bayesian procedure allows keeping the non-identifiable parameters during the uncertainty estimation since they are represented as random variables by their respective "subjective" priors.

For linear models in parameters, the covariance matrix (i. e. estimated by the FIM or Hessian inversion) fully describes the practical identifiability of model-parameters; if it has zero eigenvalues then some model-parameters are non-identifiable (Renard et al., 2010). In the wastewater community, local trajectory sensitivity analysis (i. e. based on the use of the FIM) is commonly used to assess the practical identifiability of model-parameters (Dochain and Vanrolleghem, 2001; Petersen et al., 2001; Guisasola et al., 2005). For non-linear models, near-zero eigenvalues of the covariance matrix remain only indicative of non-identifiability (Renard et al., 2010) and a local derivative-based approach may be inconclusive (Saltelli et al., 2006). As an alternative to the local trajectory sensitivity analysis, a global sensitivity analysis (GSA) approach can be applied (Sin et al., 2011). When the GSA is performed over the goodness-of-fit function (Ratto et al., 2001), it is possible to assess which model-parameters (i. e. model input factors) drive, alone or within interactions, the model fitting to data.

The aim of this chapter is to apply and compare frequentist and Bayesian inference methods to an activated sludge model, which considers simultaneous storage and growth processes for heterotrophic biomass under aerobic conditions (Sin et al., 2005a). The case study is interesting because the model-parameter inference exercise tends to be "ill-posed" if only are used OUR data.

4.3 Material and Methods

4.3.1 Global Sensitivity Analysis (GSA)

Sensitivity analysis is the study of how uncertainty in the model-output can be apportioned to different sources of uncertainty in the model input factors (Saltelli et al., 2010). Global sensitivity analysis (GSA) focuses on the output uncertainty over the entire range of possible values of the input factors. When non-linear models are considered, "variance-based" methods are well suited to account for interactions between factors. On this regard, given a model of the form $Y = f(\theta)$, with Y a scalar, a variance-based main effect for a generic factor θ_i (i = 1, ..., P) can be written as (Saltelli et al., 2010)

$$V_{\theta_i}\left(E_{\theta_{\sim i}}\left\{Y \mid \theta_i\right\}\right), \tag{4.3.1}$$

where θ_i is the i-th factor and $\theta_{\sim i}$ denotes the matrix of all factors but θ_i. The meaning of the inner expectation operator, E, is that the mean of Y is taken over all possible values of $\theta_{\sim i}$ while keeping θ_i fixed. The outer variance, V, is taken over all possible values of θ_i. When Eq. (4.3.1) is normalized by the unconditional total variance $V(Y) = V_{\theta_i}\left(E_{\theta_{\sim i}}\left\{Y \mid \theta_i\right\}\right) + E_{\theta_i}\left(V_{\theta_{\sim i}}\left\{Y \mid \theta_i\right\}\right)$, we obtain the associated sensitivity measure (main effect index, S_i) written as

$$S_i = \frac{V_{\theta_i}\left(E_{\theta_{\sim i}}\left\{Y \mid \theta_i\right\}\right)}{V(Y)}. \tag{4.3.2}$$

In a similar way, the first-order interaction effect index ($S_{i,j}$) can be written as

$$S_{i,j} = \frac{V_{\theta_{i,j}}\left(E_{\theta_{\sim i,j}}\left\{Y \mid \theta_{i,j}\right\}\right)}{V(Y)}. \tag{4.3.3}$$

Another important sensitivity measure is the total effect index (S_{Ti}), defined as

$$S_{Ti} = \frac{E_{\theta_{\sim i}} \left(V_{\theta_i} \left\{ Y \mid \theta_{\sim i} \right\} \right)}{V(Y)} = 1 - \frac{V_{X_{\sim i}} \left(E_{\theta_i} \left\{ Y \mid \theta_{\sim i} \right\} \right)}{V(Y)}, \tag{4.3.4}$$

which measures the main, first- and higher- order interaction effects of factor θ_i. In probabilistic GSA, the factor θ is a stochastic variable characterized by a distribution $g(\theta)$ that describes our prior knowledge over θ. In our case, the model input factor θ is the model-parameter to be inferred while the function $f(\theta)$ is the sum-of-squares (SS) defined in the following Sub-section 4.3.2. Intuitively, a model-parameter θ can be estimated with precision if it drives alone the model fitting to available data (i. e. S_i is high). A parameter involved in interactions (i. e. $S_{Ti}>>S_i$) that cannot be estimated with precision (i. e. S_i is low) should still be considered as important in order to achieve good fitting to data.

4.3.2 Bayesian Inference

In the Bayesian framework the concept of probability, p, is defined as the "degree of belief" or the plausibility that a proposition is true and is quantified as a real, positive number in the range of [0, 1]. Suppose we want to determine the probability of a continuous parameter, θ, given data, D, with the k-th element D_k (k = 1, ..., N), and considering the prior information on the parameter θ. Inference concerning θ is based on its posterior distribution (D'Agostini, 2003),

$$p(\theta|D) \propto p(\theta)\, p(D|\theta), \tag{4.3.5}$$

which depends both on our "subjective" belief over the parameter θ through the prior $p(\theta)$ and on data D through the likelihood $p(D|\theta)$. The prior $p(\theta)$ is formulated "before" data are observed while the likelihood $p(D|\theta)$ is a conditional probability in function of θ, with D held fixed. Note that commonly the likelihood $p(D|\theta) = L(\theta;D)$ is seen as a mathematical function of θ, with parameters D. At this point, the uncertainty in the future model predictions $y = f(\theta)$ can be inferred as:

$$p(y|D) = \int p(y, \theta|D)\, \mathrm{d}\theta = \int p(y|\theta)\, p(\theta|D)\, \mathrm{d}\theta, \tag{4.3.6}$$

where y and D are assumed to be conditionally independent given the value of θ.

A typically used measurement model for the observations D is

$$D_k = f_k(\theta) + \varepsilon, \qquad \varepsilon \sim Norm\left(0, \sigma^2\right), \tag{4.3.7}$$

where the error term ε is an independent and identically distributed (IID) random variable with a homoscedastic variance error parameter, σ^2; the corresponding likelihood for this measurement model takes the form

$$p\left(D|\theta,\sigma^2\right) \propto \exp\left\{-\frac{1}{2\sigma^2}SS\left(\theta;D\right)\right\}, \tag{4.3.8}$$

where

$$SS\left(\theta;D\right) = \sum_{k=1}^{N}\left(D_k - f_k\left(\theta\right)\right)^2 \tag{4.3.9}$$

is the sum-of-squares. For a general likelihood function, $SS(\theta;D)$ corresponds to twice the negative log-likelihood, -2ln($p(D|\theta)$) (Laine, 2008). In practice, the assumed measurement model (i. e. including error as a normal distribution -Eq. (4.3.7)- is tested against the relative distribution of residuals; if the statistical model is miss-specified, a common procedure is to evaluate how much the posterior change when other reasonable measurement models are used (Gelman et al., 2004).

Finally, note that we expressed the posterior $p(\theta|D)$ in Eq. (4.3.5) to be only proportional to the product of a given prior and likelihood. The reason is that for commonly used numerical methods the posterior is not provided in a closed form solution. For example, algorithms based on Markov chain Monte Carlo (MCMC) produce correlated samples $\{\theta^{(s)}$, s = 1, ..., $S\}$ (i. e. a Markov chain) that has the posterior $p(\theta|D)$ as its equilibrium distribution. Based on the sample $\theta^{(s)}$, the posterior is "recovered" within a histogram or kernel density approximation. Thus, the quality of such approximation improves as a function of the number of samples from the posterior (Gelman et al., 2004).

4.3.3 Frequentist Inference

Under specific assumptions, the frequentist approach is nothing but a particular case of the Bayesian inference. Assuming that a uniform distribution is a practical choice for the prior, Eq. (4.3.5) becomes (D'Agostini, 2003)

$$p\left(\theta|D\right) \propto p\left(D|\theta\right) = L\left(\theta;D\right). \tag{4.3.10}$$

The set of θ that is most likely is that which maximizes $L(\theta;D)$: a result that fits with the maximum likelihood principle. Thus, if the error ε is assumed as a normal distribution as in Eq. (4.3.7), the likelihood-dominated result becomes,

$$p\left(\theta|D\right) \propto \exp\left[-\frac{1}{2}\chi_N^2\right], \tag{4.3.11}$$

where

$$\chi^2 = \chi_N^2 = \frac{1}{\sigma^2}\sum_{k=1}^{N}\left(D_k - f_k\left(\theta\right)\right)^2 \tag{4.3.12}$$

is the chi-square statistic with N degrees of freedom. Maximizing the likelihood is equivalent to minimizing χ^2, and the most probable value of θ (i. e. the MLE) is the least-square (LS) estimate, θ_m (index "m" stands for mode). Moreover, if $f(\theta)$ is a linear model in parameters the shape of χ^2 is parabolic and thus the posterior is multi-variate Gaussian (MVG). It is then possible to estimate the uncertainty of the LS-estimate θ_m since the posterior is completely defined by its covariance matrix, $\mathbf{C}$, estimated for example from the Hessian, $\mathbf{H}$, (Marsili-Libelli et al., 2003) as

$$\mathbf{C} = \frac{SS\left(\theta_\mathrm{m}\right)}{N-P}2\mathbf{H}\left(SS\left(\theta_\mathrm{m}\right)\right)^{-1}. \tag{4.3.13}$$

Summarizing, the frequentist uncertainty estimation procedure for θ was obtained by starting from a more general framework (i. e. Bayesian framework), under clearly state hypotheses:

- The prior probability $p(\theta)$ is uniform.
- The measurement error ε is $Norm(\mathbf{0},\sigma^2\mathbf{I}_N)$, with $N\times N$ identity matrix $\mathbf{I}_N$.
- The LS-estimate θ_m is asymptotically θ^* (the true value of θ).
- $f(\theta)$ is a linear model in parameters (at least locally around θ_m).

In routine applications involving non-linear models in parameters, the linear approximation of the uncertainty of θ often holds because the above hypotheses are just slightly violated (e. g., the model is only "slightly" non-linear) or because the posterior exhibits an approximate MVG shape (e. g., large data set in relation to the number of inferred parameters).

4.3.4 Activated Sludge Model Structure

The model structure considered was the modified version of the ASM3 (Gujer et al., 1999) proposed by Sin et al. (2005a). This model includes simultaneous storage and growth processes for heterotrophic biomass and exclusively considers aerobic conditions.

Accounting for all the bioprocesses included in the model (i. e. the liquid-gas oxygen transference due to aeration is not encompassed here), the oxygen uptake rate (OUR = -dS_{O2}/dt) outcome is given by Eq. (4.3.14)

$$\begin{aligned} OUR &= \left(\frac{1-Y_{STO}}{Y_{STO}}\right)\cdot\left(1-e^{-t/\tau}\right)\cdot k_{STO}\cdot\frac{S_S}{K_S+S_S}\cdot X_H + \\ &+\left(\frac{1-Y_{H,S}}{Y_{H,S}}\right)\cdot\left(1-e^{-t/\tau}\right)\cdot\mu_{MAX,S}\cdot\frac{S_S}{K_S+S_S}\cdot X_H + \\ &+\left(\frac{1-Y_{H,STO}}{Y_{H,STO}}\right)\cdot\mu_{MAX,STO}\cdot\frac{K_S}{K_S+S_S}\cdot\frac{\left(\frac{X_{STO}}{X_H}\right)^2}{K_2+K_1\cdot\frac{X_{STO}}{X_H}}\cdot X_H + \\ &+(1-f_{XI})\cdot b_H\cdot X_H + b_{STO}\cdot X_{STO}, \end{aligned} \tag{4.3.14}$$

and the corresponding storage products production rate (SPR = dX_{STO}/dt) by Eq. (4.3.15),

$$\begin{aligned} SPR &= \left(1-e^{-t/\tau}\right)\cdot k_{STO}\cdot\frac{S_S}{K_S+S_S}\cdot X_H + \\ &-\left(\frac{1}{Y_{H,STO}}\right)\cdot\mu_{MAX,STO}\cdot\frac{K_S}{K_S+S_S}\cdot\frac{\left(\frac{X_{STO}}{X_H}\right)^2}{K_2+K_1\cdot\frac{X_{STO}}{X_H}}\cdot X_H + \\ &-b_{STO}\cdot X_{STO}, \end{aligned} \tag{4.3.15}$$

where the following nomenclature was used: S_{O2}: dissolved oxygen (DO, mg O_2/L); S_S: substrate (mg COD/L); X_H: biomass (mg COD/L); X_{STO}: storage products (mg COD/L); t: time (min); τ: first-order time constant (min); k_{STO}: maximum storage rate of X_H (1/d); K_S: S_S affinity constant (mg COD/L); K_2: a lumped parameter related to the affinity of X_H towards X_{STO}/X_H (mg COD/mg COD); K_1: regulation constant of X_H as function of X_{STO}/X_H (mg COD/mg COD); $\mu_{MAX,S}$: maximum growth rate of X_H on S_S (1/d); $\mu_{MAX,STO}$: maximum growth rate of X_H on X_{STO} (1/d); f_{XI}: production of inert COD in endogenous respiration (mg COD/mg COD); b_H: endogenous decay coefficient of X_H (1/d); b_{STO}: endogenous decay coefficient of X_{STO} (1/d); Y_{STO}: yield coefficient of X_H for storage on S_S (mg COD/mg COD); $Y_{H,S}$: yield coefficient of X_H for growth on S_S (mg COD/mg COD); $Y_{H,STO}$: yield coefficient of X_H for growth on X_{STO} (mg COD/mg COD); COD: Chemical Oxygen Demand. According to the assumptions done by Sin et al. (2005), the three yield coefficients (Y_{STO}, $Y_{H,S}$, and $Y_{H,STO}$) were considered to exclusively depend on the efficiency of the oxidative phosphorylation (δ)

(mol ATP/mol NADH$_2$), model-parameters k_{STO} and $\mu_{MAX,S}$ were considered to depend on the fraction of substrate used for storage and the maximum substrate uptake rate, and model-parameters $\mu_{MAX,S}$ and $\mu_{MAX,STO}$ were assumed equals. Furthermore, it was considered that the concentrations of DO and ammonium were high enough to not affect kinetics.

4.3.5 Model Implementation and Computational Analyses

The model was implemented in MATLAB (Mathworks Inc., USA) as a Simulink S-function Cmex code block. A free-MATLAB toolbox called "Adaptive Robust Numerical Differentiation" (D'Errico, 2006) was used for the estimation of the Hessian matrix, **H**. The estimation routine is based on a finite-difference (FD), fourth-order Romberg-extrapolation method with an adaptive routine for the determination of the step-size-perturbation parameters. The measurement model defined in Eq. (4.3.7) was considered as the reference statistical model for data. A strictly positive uniform prior were assumed in order to preserve the physical meaning of the parameters. The error variance σ^2 was considered as an unknown stochastic parameter, characterized by the inverse gamma distribution prior (Gelman et al., 2004; Laine, 2008) with mean of 0.01 and accuracy of 4. We verified that the computed posterior was insensible to the relative choice of the gamma prior parameter. The posterior $p(\theta \mid D)$ was approximated with a sample size S of 15,000 (burn-in sample of 5,000), obtained after convergence of the "Delayed Rejection Adaptive MCMC" (DRAM) sampler (Laine, 2008). The Markov chain was considered stationary if the value of the Geweke's convergence diagnostic (Geweke, 1992) was higher than 0.9.

For a one-dimensional case, the Kolmogorov-Smirnov (K-S) statistic is defined as (Peacock, 1983)

$$K{-}S = \sup_x \left| F_S^1(x_1) - F_S^2(x_1) \right|, \tag{4.3.16}$$

where K-S is used to quantify the distance between the empirical cumulative distribution function F_S^1 and F_S^2 with S IID observations from the stochastic variable X_1. In a two dimensional case (with a new stochastic variable X_2), the Peacock's algorithm considers the four quadrants ($x_1{<}X_1$, $x_2{<}X_2$), ($x_1{<}X_1$, $x_2{>}X_2$), ($x_1{>}X_1$, $x_2{<}X_2$) and ($x_1{>}X_1$, $x_2{>}X_2$) in turn, and adopt the largest of the four differences between the two empirical cumulative distributions as the final K-S statistic.

4.3.6 Data Sources

The experimental data analyzed consisted of three OUR profiles. In all cases, the initial substrate to biomass ratio was low, with feast plateaus elapsing 15 min. Feast periods were followed by famine periods; that is when OUR suddenly drops from its maximum level to a level higher than the endogenous OUR and thereafter gradually decreases to the endogenous level. For each respirogram, a sub-sampled number of measurements were considered for inference (from 1.5 to 2.8 min/sample).

Data set A was kindly provided by authors Sin et al. (2005a). It consisted in two acetate pulses of 40 mg COD/L added according to an optimal experimental design to activate sludge collected from a municipal wastewater treatment plant (WWTP) performing N-removal. The authors Sin et al. (2005a) also provided for the same experiment the off-line measurements of X_{STO} (measured as poly-β -hydroxybutyrate, PHB). $X_H(0)$ was calculated as 800 mg COD/L, and $X_{STO}(0)$ was measured as 6.8 mg COD/L.

Data set B was taken from the work of Hoque et al. (2009). It consisted in a single acetate pulse of 50 mg COD/L (pH 7.8, 20ºC) added to activated sludge collected from a WWTP. In this case, $X_H(0)$ was calculated as 900 mg COD/L.

Data set C was obtained in our lab and consisted in a single acetate pulse of 40 mg COD/L (pH 8.0, 20ºC) added to activated sludge purged from a sequencing batch reactor (SBR) fed with raw leachate under pulse feeding (4 pulses/day), loading rate of 1 g COD/(Ld), and intermittent aeration. The composition of the leachate was equivalent to 9.81 g $\mathrm{COD_{VFA}}$/L (VFA: volatile fatty acid; 25% acetate, 9% propionate, 52% butyrate, and 14% valerate) with 48% $\mathrm{COD_{VFA}}$/COD and 1.01 g N/L. The respirometric test was carried out in a 2.5-L LFS respirometer with flowing gas and static liquid (Spanjers et al., 1998). The OUR value was then estimated off-line from DO measurements (Inolab 740 - CellOx 325, WTW, Germany) by applying an optimal local polynomial filtration paradigm called “Lazy Learning” (Bontempi et al., 1997). The initial content of storage products in biomass, $X_{STO}(0)$, was assumed as 7.6 mg COD/L. Similarly to Sin et al. (2005a) and Hoque et al. (2009), once fixed parameters b_{STO} (0.2 d^{-1}), b_H (0.2 d^{-1}), and f_{XI} (0.2 mg COD·mg COD^{-1}) according to the values given in the ASM3, the initial concentration of biomass, $X_H(0)$, was calculated from the endogenous OUR as 214 mg COD/L.

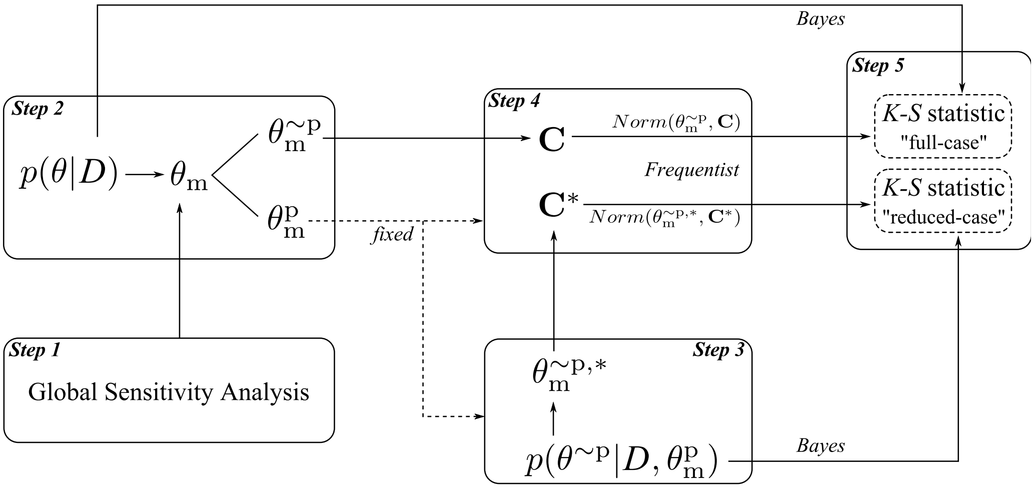

Bayes
Step 5
Step 2
Step 4
$p(\theta|D) \rightarrow \theta_{\mathrm{m}}$
$\theta_{\mathrm{m}}^{\sim\mathrm{p}}$
$\theta_{\mathrm{m}}^{\mathrm{p}}$
fixed
$\mathbf{C}$
$\mathbf{C}^*$
$Norm(\theta_{\mathrm{m}}^{\sim\mathrm{p}}, \mathbf{C})$
Frequentist
$Norm(\theta_{\mathrm{m}}^{\sim\mathrm{p},*}, \mathbf{C}^*)$
K-S statistic
"full-case"
K-S statistic
"reduced-case"
Step 1
Global Sensitivity Analysis
Step 3
$\theta_{\mathrm{m}}^{\sim\mathrm{p},*}$
$p(\theta^{\sim\mathrm{p}}|D, \theta_{\mathrm{m}}^{\mathrm{p}})$
Bayes

Figure 4.3.1: Schema of the uncertainty analysis procedure including frequentist and Bayesian inference.

4.3.7 Data Analysis Procedure

The estimated parameter vector was $\theta = [\tau\ K_S\ k_{STO}\ \mu_{MAX,S}\ \delta\ K_1\ K_2]$. Because the parameters K_1 and K_2 were found non-identifiable by Sin et al. (2005a) and Hoque et al. (2009) the parameter θ was divided in two sub-parameters: potentially “problematic” $\theta^{\mathrm{p}} = [K_1\ K_2]$ and “non-problematic” $\theta^{\sim\mathbf{p}} = [\tau\ K_S\ k_{STO}\ \mu_{MAX,S}\ \delta]$. At this point, the uncertainty analysis procedure was the following (see Figure 4.3.1):

Step 1. GSA based on the analysis of the three OUR profiles was performed by a Bayesian-GSA, free-software tool (Oakley and O’Hagan, 2004), which is able to compute the main effect index (S_i), the first-order effect index ($S_{i,j}$), and the total effect index (S_{Ti}). The distribution $g(\theta_i)$ was assumed uniform, i. e. $Unif(a_i,b_i)$, for all the model-parameters; in particular: $\tau \sim Unif(0.1, 5)$, $K_s \sim Unif(0.1, 10)$, $K_1 = Unif(10^{-2}, 1)$, $K_1 = Unif(10^{-4}, 10^{-2})$, $\delta = Unif(1, 8)$, $k_{sto} \sim Unif(0.1, 12)$, and $\mu_{MAX,S} \sim Unif(1, 20)$.

Step 2. After running the DRAM routine, a sample from the posterior $p(\theta|D)$ was obtained and $\theta_{\mathrm{m}} = [\theta_{\mathrm{m}}^{\mathrm{p}}\ \theta_{\mathrm{m}}^{\sim\mathrm{p}}]$ was calculated from the kernel-approximation of the MCMC-sample. Residuals analysis was carried out based on residuals histograms. Auto-correlation plots (Dodge, 2008) were performed in order to check the appropriateness of the measurement model used (i. e. including error as a normal distribution). The effect of the measurement model on the posterior was subsequently analyzed considering three additional error distribution alternatives: *t*-

distribution, Laplace, and Normal with weighted residuals.

Step 3. The subset θ^{p} was fixed to $\theta^{\mathrm{p}}_{\mathrm{m}}$ and the posterior $p(\theta^{\sim\mathrm{p}}|D,\theta^{\mathrm{p}}_{\mathrm{m}})$ was sampled by the DRAM algorithm. The mode-estimate $\theta^{\sim\mathrm{p},*}_{\mathrm{m}}$ of the posterior $p(\theta^{\sim\mathrm{p}}|D,\theta^{\mathrm{p}}_{\mathrm{m}})$ was calculated.

Step 4. The covariance matrix $\mathbf{C}$ for the parameter point-estimation $\theta^{\sim\mathrm{p}}_{\mathrm{m}}$ and the covariance matrix $\mathbf{C}^{*}$relative to $\theta^{\sim\mathrm{p},*}_{\mathrm{m}}$ were estimated based on Eq. (4.3.13), while θ^{p} was fixed to $\theta^{\mathrm{p}}_{\mathrm{m}}$.

Step 5. The posterior distribution $p(\theta^{\mathrm{p}}|D)$ was compared with its linear approximation, $Norm(\theta^{\sim\mathrm{p}}_{\mathrm{m}},\mathbf{C})$; for simplicity, we called this case "full-case". The "reduced-case" comparison was performed over the posterior $p(\theta^{\sim\mathrm{p}}|D,\ \theta^{\mathrm{p}}_{\mathrm{m}})$ and its linear approximation, $Norm(\theta^{\sim\mathrm{p}}_{\mathrm{m}},\mathbf{C}^{*})$. Differences between posteriors and relative approximations were evaluated with a two-sample, two-dimensional, *K-S* statistic.

4.4 Results and Discussion

4.4.1 Global Sensitivity Analysis (GSA)

The main effect index (S_i) and the total effect index (S_{Ti}) for the three data sets analyzed are reported in Table 4.1. Based on the GSA-interpretation guidelines provided by Ratto et al. (2001), we observed the following:

- δ and $\mu_{MAX,S}$ drive the model-fit to data, since they have the highest indexes S_i (δ: 53.25-58.44%, $\mu_{MAX,S}$: 4.78-14.56%) and S_{Ti} (δ: 79.23-89.91%, $\mu_{MAX,S}$: 29.32-35.71%). The majority of the interaction effects including these model-parameters is attributed to first-order interactions (i. e. $S_{i,j}$ indexes are high, with total values of 19.0-24.1%). Because these two model-parameters have relatively high main effects, both are expected to be well-estimated during the inference exercise.

- k_{STO} can be judged as the less influent model-parameter over the SS-variance, as it has the lowest S_{Ti} (0.01-0.7%), and non-identifiable because S_i (0.01-0.08%) is negligible.

- K_s and τ are almost non-identifiable (low precision of estimation), as their indexes S_i are low, leaving the main contribution to the SS-variance to the interaction terms.

- Non-identifiable, but still important parameters are K_1 and K_2.

Table 4.1: GSA sensitivity indexes for model-parameters. S_i: main effect index, S_{Ti}: total effect index. Units: %.

Parameter (θ)	Data set A[*]		Data set B		Data set C	
	S_i	S_{Ti}	S_i	S_{Ti}	S_i	S_{Ti}
τ	2.97	6.0	3.25	8.96	0.38	2.66
k_{STO}	0.01	0.01	0.08	0.08	0.06	0.70
K_S	1.24	4.0	1.23	3.45	1.21	8.12
$\mu_{MAX,S}$	14.56	32.83	13.42	29.32	4.78	35.71
K_1	0.09	2.80	0.10	4.10	0.23	7.05
K_2	0.05	2.87	0.06	3.65	0.28	5.98
δ	58.44	79.23	58.75	79.82	53.25	89.91
Total main effect[§][a]	77.4		76.9		60.2	
Total first-order inter. effect[#][b]	19.0		24.1		24.1	
Sum [a + b]	96.3		95.1		92.3	

[*] Data set A: Sin et al. (2005). Data set B: Hoque et al. (2009). Data set C: own data.

[§] Total main effect: ΣS_i

[#] Total first-order interaction effect: $\Sigma S_{i,j}$. Individual first-order interaction effect indexes are not shown.

- The high differences between S_i and S_{Ti} for a given model-parameter imply that the model is over-parameterized with respect to available OUR data. However, all the model-parameters are still necessary in order to keep the structure of the model and to reproduce data correctly.

We also observed that the condition number for **H** relative to $\theta^{\sim \mathrm{p}} = [\tau \ K_S \ k_{STO} \ \mu_{MAX,S} \ \delta]$ was 10^3 and thus, the covariance-matrix estimation can be considered as "non-problematic". On the other hand, the condition number for θ was higher than the machine precision, confirming the "problematic" nature of the parameter $\theta^{\mathrm{p}} = [K_1 \ K_2]$ for the estimation of the covariance matrix.

Based on the above considerations, we conclude that only δ and $\mu_{MAX,S}$ are expected to be well-estimated when exclusively considering OUR profiles. According to model assumptions detailed in section 2.4, appropriate estimation of δ will favor precise calculations of the three yield coefficients (Y_{STO}, $Y_{H,S}$, and $Y_{H,STO}$). Availability of additional experimental data may help in reducing differences between indexes S_i and S_{Ti}, and thus, in enhancing practical identifiability of some model-parameters.

4.4.2 Bayesian Inference

The marginal posteriors of θ_i for data sets A, B, and C are depicted in Figure 4.4.1. The relative modes θ_{m} of the data sets A and B are close to those initially reported by the

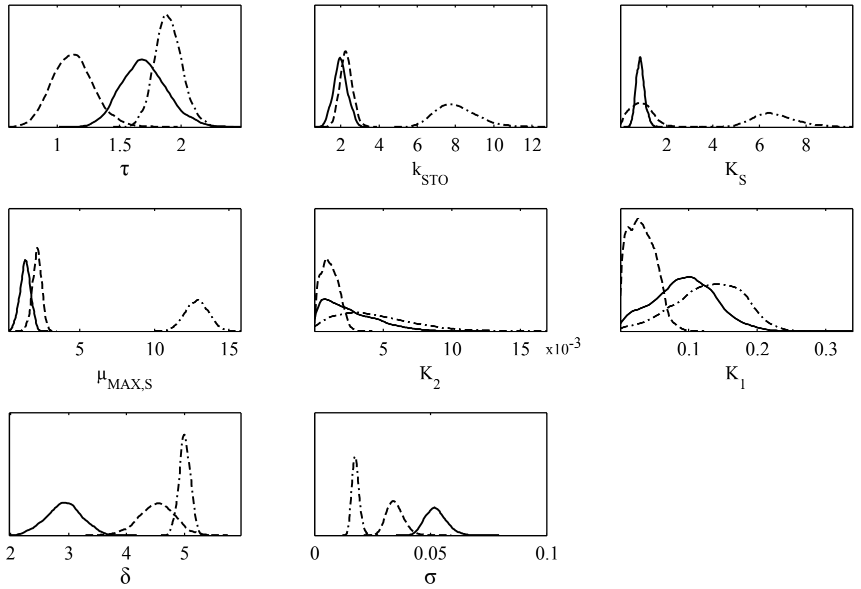


Figure 4.4.1: Marginal parameter posteriors for data set A (-), data set B (–) and data set C (- · -). Y-axis represents qualitatively the probability density.

respective authors (Sin et al., 2005a; Hoque et al., 2009).

The only exemption is the first-order time constant τ for data set A, where the mode is 1.67 ±0.36 min while the value estimated by the authors was 0.51 ±0.07 min. The difference may be caused by the sub-sampling pre-processing of the data set A, necessary to reduce the auto-correlation of the relative residuals. The MLE obtained for δ when analyzing data sets B (4.57 ±0.6) and C (5.00 ±0.2) are higher than the theoretically expectable values, of between 1 and 3 (Beun et al., 2000; Dias et al., 2008), making the mechanistic meaning of the δ-parameter questionable. Such high values of δ also lead to high yield coefficients. For all cases, Y_{STO} (0.80-0.90) is found to be the highest yield coefficient. Direct biomass formation on acetate ($Y_{H,S}$) (0.57-0.75) is estimated as very similar to the indirect biomass formation via storage products ($Y_{STO} \cdot Y_{H,STO}$) for the three data sets (within a narrow ratio between both pathways of 0.96-0.98), which is in agreement with the findings of Beun et al. (2000). Values for model-parameters k_{STO}, $\mu_{MAX,S}$ and K_S were higher for data set C than for data sets A and B. Also, inference results for data set C are the only providing k_{STO} lower than $\mu_{MAX,S}$. Thus, different values are assessed for a given model-parameter depending on the data set analyzed. This may be because model-parameter values are expected to be highly influenced by

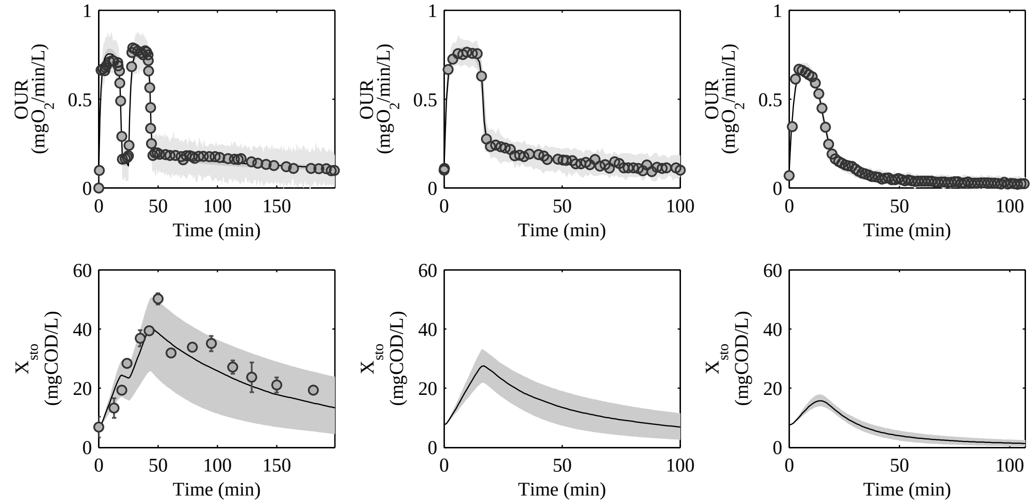


Figure 4.4.2: OUR and X_{STO} predictive envelopes (95% credibility) relative to the Bayesian inference of θ. The light-grey envelope is due to measurement errors (only for the OUR case, *upper row*) while the dark-grey envelope is due to parameter uncertainty (always wider in the case of X_{STO} than in the case of OUR). Left column for data set A, middle column for data set B, and right column for data set C.

many factors such as wastewater composition, bioreactor operating conditions, dominant microbial communities, etc. (Ni and Yu, 2008). On the other hand, boundary conditions considered may affect decisively on the results obtained through the inference exercise. This may be the case of $X_{STO}(0)$, a potential source of problems for the identification of model-parameters when using OUR alone (Sin et al., 2005a). Also, may be the case of $X_H(0)$, which was calculated taking into account the endogenous OUR and assuming the fixed values for f_{XI} and b_H proposed by Gujer et al. (1999). Any miscalculation of this boundary condition will be propagated during the inference exercise, and may result in non-realistic results.

The estimated k_{STO}-parameter uncertainty is relatively low (see Figure 4.4.1). That seems to be controversial with the GSA-result where it was found non-identifiable and un-important. Therefore, in problems where parameters have a substantial meaning, Bayesian statistics will be useful if identifiability is warranted. Otherwise, the analysis will only provide illusory solutions (San Martín and González, 2010). On this regard, if an informative prior for k_{STO} (or additional data) would be used instead of the non-informative prior we should expect that the marginal posterior of k_{STO} would be easily displaced towards the informative prior since the OUR data provide only weak information for the inference of k_{STO}. Such displacement of the marginal posterior would take place without affecting the model-fit to data since k_{STO} was found un-important with respect to model-outcomes during GSA (i. e. $S_{Ti} \leq 0.7\%$).

Predictive envelopes for the OUR and X_{STO} model-outcomes are represented in Figure 4.4.2. In the case of data set A, X_{STO}-measurements were available but not considered for the inference of parameters. Because parameter-inference was based only on OUR data sets, the OUR-predictive parameter uncertainty envelopes (Figure 4.4.2, *upper row*) were found narrower than those of the X_{STO}-envelopes (Figure 4.4.2, *lower row*). However, there is still variability in the OUR prediction (i. e. wideness of the predictive measurement envelope), especially for data set A, which might be due to measurement error or model inadequacy. The high value of σ (0.052 $\pm 9{\cdot}10^{-3}$) for data set A (see Figure 4.4.1) reflects that the model have some problems in reproducing the dynamic phase of the OUR response. Indeed, Sin et al. (2005a) already reported that the model was unable to perfectly fit the second peak in the OUR profile. In our study, the model was unable to perfectly fit the first peak. Although it seems a contradiction in the results, when considering the predictive parameter uncertainty envelope it is shown that both results are possible. Lower values of σ were obtained for data set B (0.035 $\pm 7{\cdot}10^{-3}$) and C (0.018 $\pm 3{\cdot}10^{-3}$) among other reasons because of the simpler experimental designs. It is also interesting to observe that the X_{STO}-predictive parameter uncertainty envelopes for data sets B and C are narrower than for data set A, even thought the latter was optimally designed.

4.4.3 Residuals Analysis

The validity of the above statistical inferences (frequentist or Bayesian) is dependent on the assumptions taken for the measurement model defined in Eq. (4.3.7). In Figure 4.4.3 the residuals analysis are depicted based on histograms and auto-correlation plots. The histogram of the residuals for each data set is then compared with the corresponding measurement model (Figure 4.4.3, *upper row*). The auto-correlation function of the residuals (Figure 4.4.3, *lower row*) is within the significance interval levels and thus the independence requirement is fulfilled. It seems interesting to remark that, if all the samples from a modern respirometer device (with a high frequency of sampling) were used raw and without further processing, the residuals would be highly auto-correlated, which would lead to under-estimation of parameter uncertainty. In order to avoid this situation, apart from sub-sampling, whitening and correction of the residuals by means of auto-regressive models can be applied to weight them properly (Neumann and Gujer,

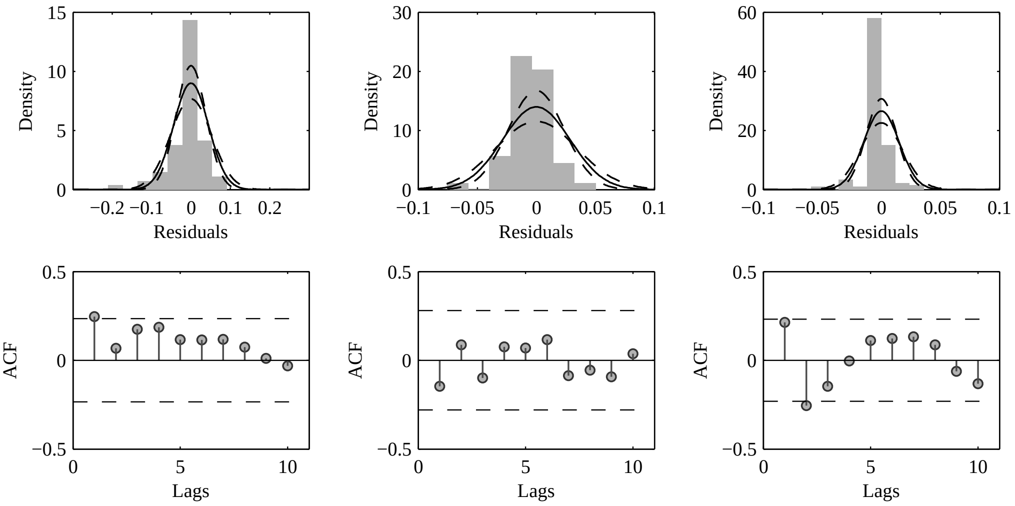


Figure 4.4.3: Residuals analysis. *Upper row*: histograms of the residuals and their relative most probable measurement models, $Norm(0,\sigma^2_{\mathrm{m}})$, in solid lines. The measurement model is given within its 95% prediction intervals in dashed lines since its parameter σ^2 is estimated during the inference of the model-parameter θ. *Lower row:* auto-correlation (ACF) plot with 95% significance intervals. Left column for data set A, middle column for data set B, and right column for data set C.

2008).

It is difficult to assess if the normality assumption over ε is appropriate or it should be rejected from the comparison between the histograms of the residuals and the corresponding measurement model (Figure 4.4.3, *upper row*). Kleinbaum et al. (2008) reported that "the confidence intervals used in a regression analysis are robust in the sense that only extreme departures of the distribution of the residuals from normality yield spurious results". In order to check the robustness of the posteriors to possible departures of the residuals from normality, in Figure 4.4.4 it is shown (adopting data set A as example) the effect of considering other measurement models rather than Normal (the reference): t-distribution with one degree-of-freedom, Laplace (or double-exponential), and Normal with weighed residuals (Normal-WLS). In the Normal-WLS case, the residuals corresponding to the feast phase were weighted in order to achieve approximate homoscedasticity. On this regard, only the marginal posterior of τ seems to be significantly affected by such election, although it is not considered as a problematic case since the parameter τ is not included in ASM models. Thus, here the posterior can be considered relatively insensitive to the particular measurement model choice. Such

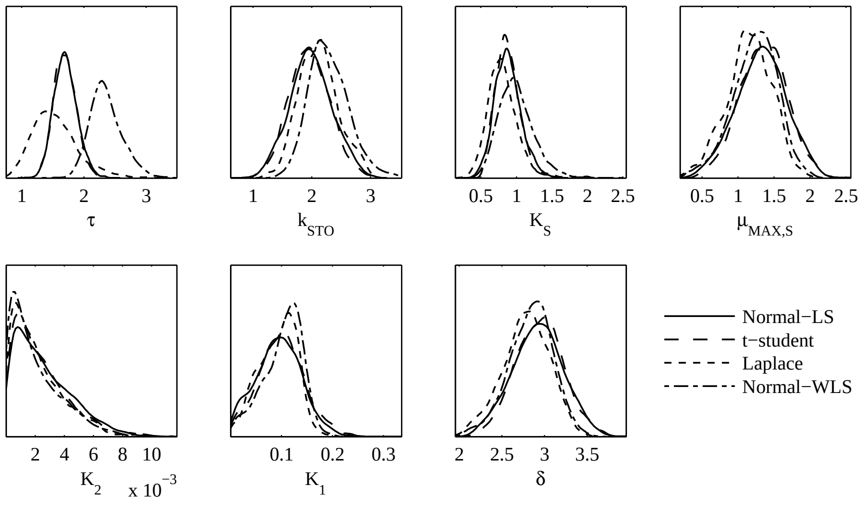


Figure 4.4.4: Effect of the measurement model on the marginal parameter posterior (data set A). Y-axis represents qualitatively the probability density.

posterior insensitivity to a particular measurement model was verified also for data sets B and C (results not shown).

It could be argued that there are a large number of measurement models applicable while we only tested four simple items. However, if the number of residuals (i. e. number of measurements) is low, the estimation of the “true” measurement distribution is difficult since any complex-enough distribution could fit the empirical histogram of residuals equally well. Thus, it seems fair to consider the Normal distribution to describe the residual-model variability when the noise process is unknown. This is because the posterior seems insensible to the model choice, making possible to maintain the normality assumption of the errors needed to proceed with the frequentist approximation from the Bayesian posterior (see Sub-section 4.3.3).

4.4.4 Comparison of Frequentist and Bayesian Inference

In order to verify the quality of the frequentist uncertainty assessment, we compared the MVG-approximation with the Bayes posterior. The comparison was performed over a two-dimensional parameter space.

For the “reduced-case” comparison, and only considering data set B as case study, we present the MCMC-sample and subsequent kernel-approximation of the posterior $p(\theta^{\sim\mathrm{p}}|D,\theta^{\mathrm{p}}_{\mathrm{m}})$, and its linear approximation $Norm(\theta^{\sim\mathrm{p}}_{\mathrm{m}},\mathbf{C}^{*})$ (Figure 4.4.5). It is necessary

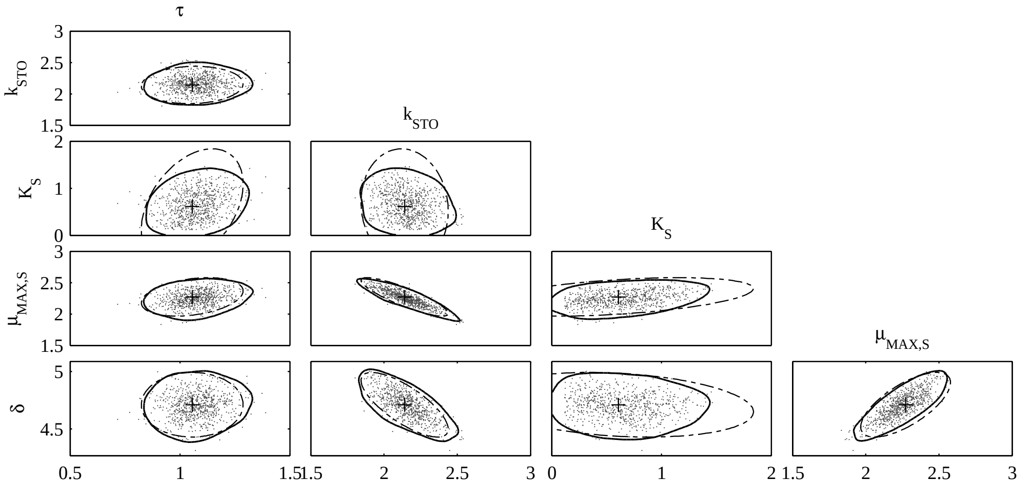


Figure 4.4.5: Two-dimensional comparison (95% credibility) between the posterior distribution $p(\theta^{\sim\mathrm{p}}|D,\theta^{\mathrm{p}}_{\mathrm{m}})$ -solid line- estimated from MCMC samples -dots- and the linear approximation $Norm(\theta^{\sim\mathrm{p}}_{\mathrm{m}},\mathbf{C}^{*})$ confidence ellipses -dashed line-. "Reduced-case" for data set B.

to remark that for this "reduced-case" the model parameter $\theta^{\mathrm{p}} = [K_1,\ K_2]$ is not considered (i. e. it is assumed as perfectly known according to authors' values) because of its multi-collinearity effect. The MVG-approximation seems to be a good approximation of the shape of the Bayesian posterior (i. e. the likelihood). However, the MVG-approximation over-estimates the uncertainty with respect to the Bayesian posterior in those cases where it is truncated as strictly positive (see for example K_S vs. k_{STO}). This is because the MVG distribution is defined over an un-bounded parameter-space. On the other hand, if it is considered the case where inference is not affected by the parameter-positivity-constrain (see for example τ vs. $\mu_{MAX,S}$), the MVG-approximation may under-estimate the parameter-uncertainty, confirming the observations of Omlin and Reichert (1999) and Vrugt and Bouten (2002). The above results were also confirmed for data sets A and C (data not shown); the MVG-approximation is a reasonable approximation for the posterior.

For both, the "reduced-case" (Figure 4.4.6, *upper row*) and the "full-case" (Figure 4.4.6, *lower row*) comparisons, and considering the three data sets, the quality of the frequentist-approximation is analyzed through the *K-S* statistics. Although for the "reduced-case" the MVG-approximation is a reasonable approximation for the posterior, for the "full-case", the same MVG-approximation is not appropriate as suggested

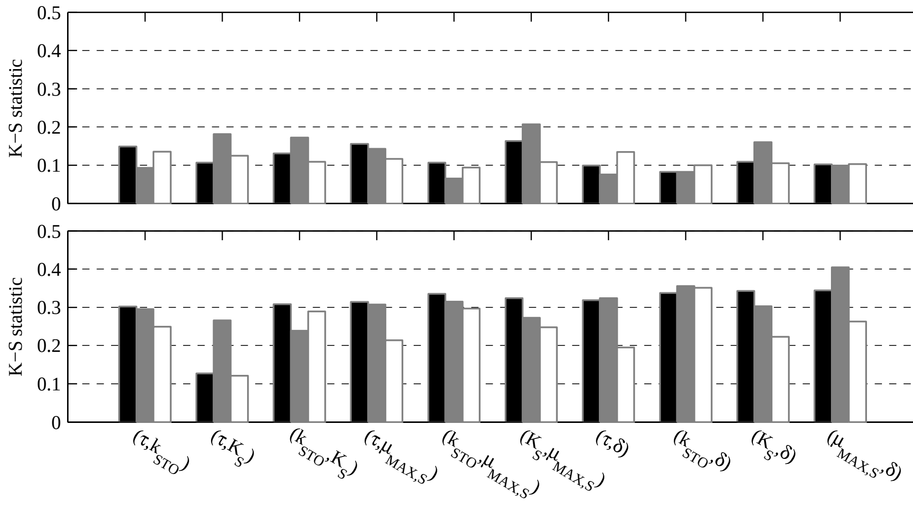


Figure 4.4.6: The two-dimensional *K-S* statistics for the “reduced-case” (*upper row*) and the “full-case” (*lower row*) comparisons relative to data set A -black bars-, data set B -grey bars- and data set C -white bars-. Figure 4.4.5 is evaluated in the *upper row* -grey color-. In the “reduced-case” the linear approximation is reasonable, while in the “full-case” it is unsatisfactory.

by the higher values of the *K-S* statistics. It is worth to remark that in the “full-case” the Bayesian posterior accounts for θ^{p} through a uniform, finite-range prior, while the corresponding MVG-approximation, $Norm(\theta^{\sim\mathrm{p}}_{\mathrm{m}},\mathbf{C})$, is estimated fixing the problematic parameter θ^{p} at $\theta^{\mathrm{p}}_{\mathrm{m}}$, the modal-values of $p(\theta|D)$. In other words, during Bayesian inference θ^{p} is weakly specified, while during the frequentist inference θ^{p} is considered as perfectly known. Note that, the assumption of a perfectly known θ^{p} is questionable (i. e. if θ^{p} is problematic because its value is highly uncertain, then how is it possible to assume it as perfectly known?) but rather necessary to make the computation of the covariance matrix feasible. Hence, the disparity in assumptions about the prior distribution of θ^{p} between Bayes and the MVG-approximation (i. e. vague vs. perfect prior knowledge) may lead to the quantitative differences observed through the *K-S* statistics. Furthermore, as the model is non-linear in parameters, the goodness of the MVG-approximation to the posterior can change just with the location of the estimation, which depends on available data.

Finally, the estimated Hessian matrix depends on the MLE obtained by a kernel-approximation of the posterior. For non-linear systems, the Hessian includes the effect of the curvature, reflecting the degree of non-linearity induced by the model structure

(Seber and Wild, 1989). The covariance matrix based on the Hessian is a linear approximation to the likelihood only when the curvature effect vanishes in the neighborhood of the MLE. Thus, it is important to reach the MLE to have a reliable linear approximation of the likelihood surface. In our case, we run a large number of MCMC samples in order to achieve a reliable MLE. However, as it has been shown above, this is a necessary but still not sufficient condition to be satisfied in order to have a reliable linear approximation of the parameter uncertainty estimation.

4.5 Conclusions

Parameter uncertainty estimation for an activated sludge model (Sin et al., 2005a) was addressed exclusively based on the analysis of (three) OUR profiles within Bayesian and frequentist inference frameworks. It was discussed how under particular hypotheses, Bayesian inference can be reduced to a frequentist derivation. Hence, comparing the results of the two inferential procedures it is possible to test if the assumed hypotheses that lead to the linear (i. e. frequentist) approximation formulae are justified. Global sensitivity analysis helped in elucidating best identifiable parameters. Effect of the measurement model choice on the Bayesian posteriors was also assessed. For the model and data at hand, only two model-parameters, δ and $\mu_{MAX,S}$, could be estimated with appropriate precision. We found that the goodness of the frequentist approximation enhanced when non-identifiable parameters where assumed as perfectly known. This is a quite common procedure in wastewater treatment modelling with sparse data. However, although this "perfect-knowledge" assumption may be questionable in many situations, it is still necessary for the computation of the frequentist-approximation. On the other hand, Bayesian inference was shown as a more flexible methodology since it allowed reasonable probabilistic description of the problematic parameter. Moreover, as there are many situations where a priori it is not possible to assess suitability of the linear approximation methods for addressing the estimation of parameter uncertainty when working with ASM-type models, it may result advisable to use Bayesian inference instead.

Acknowledgements

This research was supported by ACC1Ó/Generalitat de Catalunya under programme INNOEMPRESA-2009 (YT-CONIT project IEINN09-1-0180) and the Spanish Ministry of Science and Innovation (ADAMOX project CTM2010-18212). Authors thank Dr. A. Guisasola (UAB) for providing data concerning OUR and X_{STO} used as data set A.

Bibliography

Beun, J. J., Paletta, F., van Loosdrecht, M. C. M., Heijnen, J. J., 2000. Stoichiometry and kinetics of poly-beta-hydroxybutyrate metabolism in aerobic, slow growing, activated sludge cultures. Biotechnology and Bioengineering 67 (4), 379–389.

Bontempi, G., Birattari, M., Bersini, H., 1997. Lazy learning for local modelling and control design. International Journal of Control 72, 643–658.

Checchi, N., Marsili-Libelli, S., 2005. Reliability of parameter estimation in respirometric models. Water Research 39 (15), 3686–3696.

D'Agostini, G., 2003. Bayesian inference in processing experimental data: Principles and basic applications. Reports on Progress in Physics 66 (9), 1383–1419.

D'Errico, J., 2006. Adaptive robust numerical differentiation, matlab file exchange (accessed November 2011).
URL `http://www.mathworks.com/matlabcentral/fileexchange/13490-adaptive-robust-numerical-differentiation`

Dias, J., Oehmen, A., Serafim, L., Lemos, P., Reis, M., Oliveira, R., 2008. Metabolic modelling of polyhydroxyalkanoate copolymers production by mixed microbial cultures. BMC Systems Biology 2 (1), 59.

Dochain, D., Vanrolleghem, P. A., 2001. Dynamical Modelling and Estimation in Wastewater Treatment Processes. IWA Publishing, UK.

Dodge, Y., 2008. The Concise Encyclopedia of Statistics. Springer Science and Business Media LLC, USA.

Gelman, A., Carlin, J. B., Stern, H. S., Rubin, D. B., 2004. Bayesian Data Analysis, Texts in Statistical Science, 2nd Edition. Chapman & Hall/CRC, USA.

Geweke, J., 1992. Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. In Bayesian Statistics 4 (ed. Bernado, J. M., Berger, J. O., Dawid, A. P., Smith, A. F. M.). Oxford University Press, UK.

Guisasola, A., Sin, G., Baeza, J. A., Carrera, J., Vanrolleghem, P. A., 2005. Limitations of ASM1 and ASM3: a comparison based on batch oxygen uptake rate profiles from different full-scale wastewater treatment plants. Water Science and Technology 52 (10-11), 69–77.

Gujer, W., Henze, M., Mino, T., van Loosdrecht, M., 1999. Activated sludge model no.3. Water Science and Technology 39 (1), 183–193.

Hoque, M. A., Aravinthan, V., Pradhan, N. M., 2009. Assessment on activated sludge models for acetate biodegradation under aerobic conditions. Water Science and Technology 60 (4), 983–994.

Kleinbaum, D. G., Kupper, L. L., Nizam, A., Muller, K. E., 2008. Applied Regression Analysis and Multivariable Methods, 4th Edition. Duxbury Press, USA, page 120.

Laine, M., 2008. Adaptive MCMC methods with applications in environmental and geophysical models. Ph.D. thesis, Department Of Mathematics And Physics; Lappeenranta University Of Technology Lappeenranta.

Magrí, A., Flotats, X., 2008. Modelling of biological nitrogen removal from the liquid fraction of pig slurry in a sequencing batch reactor. Biosystems Engineering 101 (2), 239–259.

Marsili-Libelli, S., Guerrizio, S., Checchi, N., 2003. Confidence regions of estimated parameters for ecological systems. Ecological Modelling 165 (2-3), 127–146.

Marsili-Libelli, S., Tabani, F., 2002. Accuracy analysis of a respirometer for activated sludge dynamic modelling. Water Research 36 (5), 1181–1192.

Neumann, M. B., Gujer, W., 2008. Underestimation of uncertainty in statistical regression of environmental models: influence of model structure uncertainty. Environmental Science and Technology 42 (11), 4037–4043.

Ni, B.-J., Yu, H.-Q., 2008. Simulation of heterotrophic storage and growth processes in activated sludge under aerobic conditions. Chemical Engineering Journal 140 (1-3), 101–109.

Oakley, J. E., O'Hagan, A., 2004. Probabilistic sensitivity analysis of complex models: a Bayesian approach. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66 (3), 751–769.

Omlin, M., Reichert, P., 1999. A comparison of techniques for the estimation of model prediction uncertainty. Ecological Modelling 115 (1), 45–59.

Peacock, J. A., 1983. Two-dimensional goodness-of-fit testing in astronomy. Monthly Notices of the Royal Astronomy Society 202, 615–627.

Petersen, B., Gernaey, K., Vanrolleghem, P. A., 2001. Practical identifiability of model parameters by combined respirometric-titrimetric measurements. Water Science and Technology 43 (7), 347–355.

Ratto, M., Tarantola, S., Saltelli, A., 2001. Sensitivity analysis in model calibration: GSA-GLUE approach. Computer Physics Communications 136 (3), 212–224.

Renard, B., Kavetski, D., Kuczera, G., Thyer, M., Franks, S. W., 2010. Understanding predictive uncertainty in hydrologic modeling: The challenge of identifying input and structural errors. Water Resources Research 46, 22.

Saltelli, A., Annoni, P., Azzini, I., Campolongo, F., Ratto, M., Tarantola, S., 2010. Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index. Computer Physics Communications 181 (2), 259–270.

Saltelli, A., Ratto, M., Tarantola, S., Campolongo, F., 2006. Sensitivity analysis practices: Strategies for model-based inference. Reliability Engineering & System Safety 91 (10-11), 1109–1125.

San Martín, E., González, J., 2010. Bayesian identifiability: Contributions to an inconclusive debate. Chilean Journal of Statistics 1 (2), 69–91.

Seber, G. A. F., Wild, C. J., 1989. Nonlinear Regression. John Wiley & Sons Ltd., USA.

Sin, G., Gernaey, K. V., Neumann, M. B., van Loosdrech, M., Gujer, W., 2011. Global sensitivity analysis in wastewater treatment plant model applications: Prioritizing sources of uncertainty. Water Research 45 (2), 639–651.

Sin, G., Guisasola, A., De Pauw, D. J. W., Baeza, J. A., Carrera, J., Vanrolleghem, P. A., 2005a. A new approach for modelling simultaneous storage and growth processes for activated sludge systems under aerobic conditions. Biotechnology and Bioengineering 92 (5), 600–613.

Spanjers, H., Vanrolleghem, P. A., Olsson, G., Dold, P. L., 1998. Respirometry in control of the activated sludge process: Principles. Tech. rep., IAWQ. Scientific and Technical Report 7, UK.

Vanrolleghem, P. A., Spanjers, H., Petersen, B., Ginestet, P., Takacs, I., 1999. Estimating (combinations of) activated sludge model no. 1 parameters and components by respirometry. Water Science and Technology 39 (1), 195–214.

Vrugt, J. A., Bouten, W., 2002. Validity of first-order approximations to describe parameter uncertainty in soil hydrologic models. Soil Science Society of America Journal 66, 1740–1751.

5 Multi-Criteria Analyses under Uncertainty and Multiplicity

Title: Multi-criteria analyses of wastewater treatment bio-processes under an uncertainty and a multiplicity of steady states (in revision at Water Research)

Živko Južnič-Zonta[b,*], Juš Kocijan[c,d], Xavier Flotats[a,b]and Darko Vrečko[c]

[a]GIRO Technological Centre. Rambla Pompeu Fabra 1, 08100 Mollet del Vallès, Barcelona, Spain
[b]Department of Agrifood Engineering and Biotechnology. Universitat Politècnica de Catalunya. Campus del Baix Llobregat, Edifici D4, Esteve Terradas 8, 08860 Castelldefels, Barcelona, Spain
[c]Department of Systems and Control, Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
[d]School of Engineering and Management, University of Nova Gorica, Vipavska 13, 5000 Nova Gorica, Slovenia
[*]Corresponding author

5.1 Abstract

This chapter presents a multi-criteria evaluation methodology for determining the operating strategies for bio-chemical, wastewater treatment plants based on a model analysis under an uncertainty that can present multiple steady states. The method is based on Monte Carlo (MC) simulations and the expected utility theory in order to deal with the analysis of choices among risky operating strategies with multi-dimensional outcomes. The motivation is given by a case study using an anaerobic digestion model (ADM) adapted for multiple co-substrates. It is shown how the multi-criteria analyses' computational complexity can be reduced within an approximation based on Gaussian-process regression and how a reliability map can be built for a bio-process model under uncertainty and multiplicity. In our uncertainty-analyses case study, the reliability map shows the probability of a biogas-production collapse for a given set of substrates mixture input loads.

Nomenclature and Notations

X_1	Pig manure substrate inflow COD concentration (kgCOD/m^3)
X_2	Beet energy crop co-substrate inflow COD concentration (kgCOD/m^3)
k_{dis}	First-order disintegration rates (1/d)
$K_{I,NH3}$	Free-ammonia inhibition constant (kmoleN/m^3)
x_{ch}	Inflow COD fraction of carbohydrate (-)
x_{pr}	Inflow COD fraction of protein (-)
x_{li}	Inflow COD fraction of lipids (-)
x_{inert}	Inflow COD fraction of inerts (-)
$\mu_{max,ac}$	Acetoclastic maximum rate (1/d)
$X_{ac}(0)$	Initial acetoclastic population (kgCOD/m^3)
S	Number of days for the dynamic start-up period (d) - t = 1,. . . ,S
T	Number of days of model-simulation (d) - index t = *1*,. . . ,T
M	Number of uncertain inputs sampled from probability models - index j = 1,. . . ,M
Z	Number of criteria (or objectives) to be optimized - index z = 1,. . . ,Z
E	Number of steady-states (equilibria) - index i = 1,. . . ,E
E_{max}	Maximum number of steady-states possible for a fixed uncertainty analysis framing
N	Number of feasible actions - index k = 1,. . . ,N
D	Number of manipulative (or control) variables - index d = 1,. . . ,D
Q_1	Pig-manure substrate inflow (m^3/d)

Q_2	Beet energy crop co-substrate inflow (m^3/d)

5.2 Introduction

Nowadays, most wastewater treatment bio-process operators still find it difficult to select a compromise between a high-efficiency process performance and operational safety. However, a very common solution to this decision-making problem is to perform an expensive and long-lasting pilot-plant experiment. One of the reasons why plant operators trust the experimental method is that it implicitly takes into account the uncertainty inherent to the bio-chemical system. On the other hand, an outcome from the system's mechanistic model is distrusted since it fails to represent the epistemic uncertainties explicitly. Nevertheless, in many areas of wastewater treatment science and technology, mathematical models are built to simulate complex processes in order to find appropriate operating strategies. Since the process of modeling real-world phenomena is more or less biased by our epistemic uncertainty, scenario analyses by means of Monte Carlo (MC) simulations for multi-criteria evaluations are attracting increasing attention amongst the users of such “uncertain” models.

Recently, model-parameter uncertainty was included for multi-criteria evaluations of waste-water treatment plants (WWTPs) operating under different control strategies (Flores-Alsina et al., 2008; Benedetti et al., 2010) and different uncertainty analyses (UA) scenarios (Sin et al., 2009). Under uncertainty, the criterion is a stochastic quantity, since the uncertainty in the model parameters is propagated through the computer code. A limitation of the above-proposed UA is that the probability density function (PDF) that generated the criterion is constrained to be uni-modal. In other words, the UA framing should be such that only one fixed-point, steady-states solution (or equilibria) is possible.

However, it is well known that a multiplicity of steady states is very common in chemical and biological systems. Isothermal multiplicity that is the result of non-monotonic kinetics occurs only when the non-monotonic kinetic dependence of the rate of the reaction upon the species concentration is sharp enough (Elnashaie et al., 2007). For example, the WWTP models ASM (Henze et al., 2000) and ADM1 (Bastone et al., 2002) are known to have non-monotonic kinetics due to multiplicative inhibition.

Recently, stability, phase-state and bifurcation analyses were proposed in order to determine the appropriate operating-condition strategies for anaerobic digestion (AD) bio-reactors (Shen et al., 2007; Volcke et al., 2010; Sbarciog et al., 2010). Simple control laws based on this type of analysis are attractive because of their ease of implementation and the guaranteed process stability. Unfortunately, however, some difficulties arise if a real-scale, co-digestion plant that is processing complex substrates is considered. For example, the influent composition can hardly be considered as a time-invariant “parameter” of the system (e. g., manure composition depends on the animal diets, unknown dilution factors, aging times, storage conditions and other environmental disturbances). Moreover, the stability, phase state and bifurcation analyses suffer from dimensional complexity and thus only simplified models or models with a very limited number of uncertain parameters can be used.

This chapter proposes a multi-criteria evaluation methodology that can deal with a multiplicity of steady states and have no restrictions on the type of uncertainty description. Full environmental-system models can be used. The main idea presented in this contribution is: i) the multi-modal PDF of the multi-criteria evaluations is approximated with a Gaussian mixture model (GMM) and ii) its expected utility PDF is computed in order to reduce the multi-modal, multi-criteria problem to a standard uni-modal, multi-criteria problem. In order to reduce the UA’s computational complexity, the Pareto front is estimated by an approximation within the Gaussian process regression (GPR).

The chapter is organized as follows. The AD plant case study, its process model and the UA framing is presented in Section 5.3, where the GMM and the GPR are also introduced as methods used during the proposed decision-making methodology. In Section 5.4 we present the results of the decision-making methodology applied to the AD case study. The results are discussed and observations about the proposed methodology are made in Section 5.5. Finally, in Section 5.6, our conclusions and future research directions are presented.

5.3 Methods

In this chapter an anaerobic, co-digestion, biogas plant is chosen in order to test the decision-making methodology proposed in the following. The uncertainty in the influent-substrate concentration profiles is modeled with stochastic time-series surrogates, while the uncertainty of the most influential model parameters is described within PDF models. The parameter correlations are accounted for. The problem is interesting because every increase in the co-substrate’s concentration and dilution rates increases the biogas production, but inevitably deteriorates the system’s robustness (Shen et al., 2007). This sort of conflict gives rise to the system’s multiplicity. We show how it is possible to find an optimal trade-off between robustness and biogas productivity under the uncertainty for a “real” case scenario.

5.3.1 AD Case Study Used as a Reference

A full-scale mesophilic AD plant feed with pig manure (SAVA, Miralcamp, Lleida, Spain) was taken as a reference in this work. The total liquid volume of the anaerobic reactors was of 6000 m^3, with an average hydraulic retention time (HRT) of 20 days. The data were collected on a daily basis during a period of 472 days. The available data included the pH, the bio-gas outflow and the pig-manure inflow/outflow COD concentrations. Thus, the mean COD value for the inflow, X_{c1}, was 43 kgCOD/m^3. A beet energy crop was considered as a potential co-substrate to be fed into the plant. Its mean inflow COD concentration X_{c2} was 244 kgCOD/m^3. According to the methodology used by (Galí et al., 2009), the first-order disintegration kinetic constant for the pig manure was found to be 0.18 1/d, while for the beet energy crop it was 0.47 1/d. The inflow COD fractions of the substrates (reported in Sub-section 5.3.3, Table 5.2) were estimated from the substrates' characterization (Galí et al., 2009).

5.3.2 Process Simulation

The ADM1 model (Bastone et al., 2002) was adapted to consider the digestion of multiple co-substrates (Galí et al., 2009); for each co-substrate, a disintegration process with its relative COD fractions were added (inorganic carbon and nitrogen balance was adjusted accordingly). According to the methodology used by Galí et al. (2009), the first-order disintegration rates (k_{dis}) for the pig manure was found to be 0.18 1/d, while for the beet energy crop it was 0.47 1/d. The inflow COD fractions of the substrates (reported in Sub-section 5.3.3, Table 5.2) were estimated from the substrates' characterization (Galí et al., 2009).

The model was implemented in MATLAB (Mathworks Inc., USA) as a Cmex-S-function block in the Simulink simulation environment and a stiff ODE solver was used (ode15s) for its simulation. The model code was validated against the ADM1/ode model of the benchmark simulation framework BSM2 (Rosen et al., 2006). If not specified otherwise, the ADM1 parameters were held at their nominal values, as defined in Rosen and Jeppsson (2006). The sources of inorganic carbon and nitrogen were assumed to be proportional to the pig-manure inflow COD concentration: the assumed proportionality constants were 0.001 kmoleC/kgCOD and 0.0039 kmoleN/kgCOD, respectively. Since a wide range of inhibiting ammonia concentrations has been reported in the literature (Chen et al. (2008) and references therein) the free-ammonia inhibition constant $K_{I,NH3}$ was calibrated by minimizing the sum of the squared errors of the total COD outflow concentrations and the pH, considering that ammonia is the main inhibitor for this substrate. The estimated value of $K_{I,NH3}$ was 0.0027 kmoleN/m^3. This value is higher then the nominal value of 0.0018 kmoleN/m^3, as given in Rosen and Jeppsson (2006). This result is probably due to a microorganism selection mechanism since the SAVA plant has been working for more than two years with a rich-ammonia substrate.

5.3.3 Model Input Uncertainty

The collection of all the sources of uncertainty is generally called the “input uncertainty”. Two sources of input uncertainty were considered in our case: i) the model parameters and ii) the time-variable composition in the COD substrates.

Model Parameters

In the case of the ADM1 co-digestion model the parameters can be grouped into three categories: i) parameters describing the complex nature of the inflow organic substrates (e. g., the substrate disintegration kinetics, COD fractions, etc.); ii) parameters linked to the bio-chemical processes (e. g., the hydrolysis rate, acidogenesis rate, etc.), the plant's design and operation (e. g., the reactor volume, temperature, etc.), and the transfer of phases (e. g., Henry constants, etc.); and iii) the parameters related to the initial state conditions of the system (e. g., the microorganism concentrations, etc.). In our case, 168 model parameters were present in the adjusted ADM1 model for co-digestion. Because there were too many potential sources of model-parameter uncertainty, only the influential over the variance of methane production were selected. First, a sub-set of 21 model-parameters were selected based on our knowledge of the process. Then, over this sub-set, we performed a variance-based, global sensitivity analysis (SA). The global SA was based on a Bayesian procedure developed by Oakley and O'Hagan (2004), which is available as free-software called “Gaussian Emulation Machine for SA” (GEM-SA). The advantage of this probabilistic SA is its computational efficiency if compared to other standard global SA methods. In order to keep this study focused on the UA-based multi-criteria methodology, we present only SA-results. We observed, that when the inflow substrate loads were included in the SA as uncertain inputs, the most influential parameters were the disintegration rates (k_{dis}), the inflow COD fractions (carbohydrate x_{ch}, protein x_{pr}, lipids x_{li} and inert fraction x_{inert}), the acetoclastic maximum growth rate ($\mu_{max,ac}$) and the initial acetoclastic population, $X_{ac}(0)$. Moreover, we performed a sequential SA over the 21 model-parameters for a fixed value of the pig manure loading inflow (300 m^3/d) and a variable co-substrate COD loading. As expected, the result was that the $\mu_{max,ac}$ and $X_{ac}(0)$ importance increase with the increase of the co-substrate

COD loading since the bottleneck acetogenic/methanogenic reaction becomes active and methane production starts to decline. Finally, only the influent COD fractions and inflow COD concentrations relative to the substrates were assumed uncertain. The reason is two-fold: first, one should try to keep computational demand of UA low and second, because of presentation simplicity since our aim is not to achieve a high fidelity analysis but an overview of the proposed UA-methodology.

The input uncertainty is modeled within probability distributions (O'Hagan and Oakley, 2004). In our case, a Beta probability distribution was used to model the uncertainty of the inflow COD fractions (carbohydrate x_{ch}, protein x_{pr}, and lipids x_{li}) for the respective substrates. The Beta distribution is defined as

$$Beta\,(x|\alpha,\beta) = \frac{\Gamma\,(\alpha+\beta)}{\Gamma\,(\alpha)\,\Gamma\,(\beta)} x^{\alpha-1}\,(1-x)^{\beta-1}\,, \tag{5.3.1}$$

where Γ is the gamma function Γ(n) = (n-1)!. It is defined on the finite interval range (0, 1) and has only two parameters (α, β). The parameters (α, β) are related to the Beta distribution's expected value μ_b= E(x) and the variance σ_b^2 = Var(x) within the following expressions:

$$\begin{array}{rcl} \alpha & = & -\left(\mu_b^3 - \mu_b^2 + \mu_b\sigma_b^2\right)/\sigma_b^2, \\ \beta & = & \left(\mu_b - \sigma_b^2 + \mu_b\sigma_b^2 - 2\mu_b^2 + \mu_b^3\right)/\sigma_b^2. \end{array} \tag{5.3.2}$$

The Beta distribution was preferred to the commonly used uniform distribution because in our case it was unrealistic, since experts can only specify intervals. Because the Beta distribution has a wide range of shapes (i. e. symmetric and asymmetric) over a finite range of values, but few free-parameters, the expert's knowledge about the parameter-uncertainty can be easily modeled. For example, as the Beta distribution, inflow COD fraction values are defined on the finite interval range [0, 1]. Moreover, the exponential, uniform and gamma distributions are all special cases of the Beta distribution and the normal distribution can be well-approximated by the Beta distribution.

The expected values of the inflow COD fractions μ_b were assumed to be equal to the values obtained as part of the laboratory substrate characterization (Table 5.2). The uncertainty about the precision of the laboratory methods was expressed, based on the expert's data and experience, within the standard deviation measure σ_b, as given in Table 5.2, for the respective inflow substrates. The total inert fraction x_{inert} was estimated with the correlation function

$$x_{inert} = 1 - (x_{ch} + x_{pr} + x_{li}). \tag{5.3.3}$$

Table 5.2: Pig-manure waste substrate and beet energy crop co-substrate inflow COD fractions.

Substrate	Influent COD fractions (kgCOD/kgCOD)		
	carbohydrate x_{ch}	protein x_{pr}	lipids x_{li}
Pig manure μ_b	0.41	0.20	0.03
Pig manure σ_b^2	0.1	0.1	0.05
Beet μ_b	0.83	0.08	0.03
Beet σ_b^2	0.1	0.05	0.05

The sampling of COD fractions from the distributions was performed by the inverse method; the random numbers were generated from a Hammersley low-discrepancy sequence (Hammersley, 1960), since it is known to perform better than Latin hypercube sample generators for multi-dimensional MC routines (Wang et al., 2004 and therein references). Because of the uncertainty propagation associated with Eq. (5.3.3) it was necessary to apply a simple "acceptance-rejection" method in order to obtain positive values for the inert fraction, $x_{inert} \geq 0$. The ADM1 model distinguishes between the soluble and particulate inert COD fractions. It was assumed that 50% of the total inert fraction was composed of soluble inerts. For our applications, this assumption was arbitrary.

Time-Variable Composition in the COD Substrates

The second source of uncertainty was the inflow COD concentrations of the substrates $\mathrm{X_{c1}}$ and $\mathrm{X_{c2}}$. A common procedure in the WWTP simulations is to build an explicit time-series model in order to simulate the COD inflow concentrations and loads (Gernaey et al., 2011). The uncertainty is introduced when MC samples of the time-series model parameters are taken. The problem with an explicit time-series model is that there is some ambiguity in selecting the proper model class or order. Instead, we applied an alternative approach, known as the "surrogate time series", which is based on the generation of constrained randomizations of the time series of the data. The advantage of the surrogate model is that there are no parameters at all: the dynamic sample is constructed directly from the original measured data by the amplitude-adjusted, Fourier-transform, re-sampling technique (Schreiber and Schmitz, 2000). The generated random surrogates are constrained to have the same auto-correlation and the same probability distribution as the original data sets. In a multi-variate case, the cross-correlation function between the data channels is preserved. An interesting feature of this re-sampling technique is that it allows the construction of surrogates from multivariate, spiky-data time series, which is useful when simulating unexpected extreme events.

The surrogate time series were constructed from the measured pig-manure COD concentrations at the bio-gas plant of SAVA. Time-series COD measurements for the beet energy crop were not available because it was proposed as a potential co-substrate for the SAVA plant. Thus, its COD time-series concentration surrogate was assumed to have the same frequency characteristics as the pig-manure disturbance. However, the expected COD value of the beet energy crop's co-substrate was fixed to the value estimated during the substrate characterization (244 kgCOD/m^3), while its standard deviation was assumed to be 10 kgCOD/m^3. The steady-state response of the model cannot be reached in a strict sense because the COD composition from the surrogate models is a stationary stochastic process. However, after a dynamic start-up period of S-days, the mean and the variance of the anaerobic model outcomes are constant. Thus, if T is the total number of model-simulation days, we can assert that the quasi steady-state is reached over the time period of interest of T-S days.

Finally, the two sources of input uncertainty are joined in a set $\{\{x_{ch},\ x_{pr},\ x_{li},\ x_{inert}\}_{1,2}, X_1,\ X_2\}_j$ $(j = 1, \ldots, M)$, where M is the number of uncertain inputs to be taken from the probability models.

5.3.4 Gaussian Mixture Model (GMM)

If the system presents multiplicity for a given UA scenario then E groups of discrete data are possible within an overall Z-dimensional criteria data set $\mathbf{J} = \{\mathbf{J}_z : z = 1, \ldots, Z\}$. Thus, the i-th group ($i = 1, \ldots, E$) is directly associated with a particular equilibrium of the system. Summarizing, a particular value of $\mathbf{J}$ for the z-th criteria, i-th equilibria, j-th uncertain input and t-th simulation time is $J_{z,i,j,t}$; as an example of compact notation, $\mathbf{J}_{z,i,j} = \{J_{z,i,j,t} : t = S, \ldots, T\}$. A convenient way to model this data is by mixture models (McLachlan and Peel, 2000). The mixture-model parametric formulation for $\mathbf{J}$ can be expressed as

$$g\left(\mathbf{J}\right) = \sum_{i=1}^{E} \pi_i g_i\left(\mathbf{J}|\boldsymbol{\theta}_i\right), \tag{5.3.4}$$

where $g_i(\mathbf{J}|\boldsymbol{\theta}_i)$ are the conditional probability densities weighted with the mixing proportions π_i (which should sum to one) and $\boldsymbol{\theta}_i$ is the vector of unknown parameters for the i-th component density in the mixture. The probability operator $(\cdot|\cdot)$ is a conditional operator: the probability of the term on the left-hand side is conditional on the value or distribution of the term on the right-hand side. The resulting function $g(\mathbf{J})$

is a probability density from observing the overall data $\mathbf{J}$. Many types of conditional probability densities are possible: Binomial, Poisson, Exponential, etc. In our case, a Gaussian mixture model (GMM) with normal multivariate components was used

$$g_i\left(\mathbf{J}|\boldsymbol{\theta}_i\right) = \frac{1}{(2\pi)^{\frac{Z}{2}}\left|\boldsymbol{\Sigma}\right|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\left(\mathbf{J}-\mu_i\right)' \boldsymbol{\Sigma}_i^{-1}\left(\mathbf{J}-\mu_i\right)\right\}, \tag{5.3.5}$$

where $\boldsymbol{\theta}_i = [\mu_i\ \boldsymbol{\Sigma}_i]$ with a mean (vector) μ_i and a covariance matrix $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}(i = 1,\ldots,E)$. After the value of E groups is specified, the parameters $\boldsymbol{\theta}_i$ and π_i of the GMM can be inferred by an expectation-maximization algorithm as described in McLachlan and Peel (2000). The problem is how to assign the value of E. One possible solution is to observe that the multivariate normal components of GMM can represent clusters. If the maximum value of E (E_{max}) is known a priori we can experiment with a range of E values (i. e. $E = 2,\ldots,E_{max}$) and choose the GMM model with the highest average silhouette value, s (Rousseeuw, 1987). In short, the silhouette value, $-1 \leq s \leq 1$, is used to validate clustered data: when s is close to 1, data are "well-clustered", while when s is about zero, the clusters are "undistinguishable". The case where s is close to -1 is when data has been "misclassified". When only one GMM component is present ($E = 1$), s is arbitrary. If there are no clearly distinguishable clusters, i. e. s is lower then a threshold value s^*, then a GMM model with only one component ($E = 1$) is a reasonable choice in order to model the overall data set $\mathbf{J}$.

5.3.5 Multi-Criteria Evaluation Methodology

In the AD operating practice it is interesting to maximize two criteria ($Z = 2$): the methane production efficiency, $\mathbf{J}_{1(CH4)}$, defined as the methane flow per unit of reactor volume; and the COD removal efficiency, $\mathbf{J}_{2(CODrem)}$, defined as the COD removed from the reactor per COD entered with the inflow. In other words, the decision-maker objective is to choose from a set of actions $\mathbf{A} = \{\mathbf{a}_k : k = 1,\ldots,N\}$ the k-th D-dimensional action vector, $\mathbf{a}_k$, which sets the multi-criteria set $\mathbf{J}(\mathbf{a}_k) = \mathbf{J}_k$ on the Pareto front. For convenience, we called $\mathbf{A}$ the "action grid".

In our case, the action $\mathbf{a} = [\mathrm{Q}_1\ \mathrm{Q}_2]$ consisted of fixing the appropriate pig-manure inflow Q_1 and the beet energy crop inflow Q_2 rates ($D = 2$). Thus, we assumed that actions are not uncertain, but perfectly under control. The admissible actions were taken from a square region: $\mathbf{A} = \{Q_1 : [0,\ 1000],\ Q_2 : [0,\ 250]\}$ measured in m^3 of inflow per day. The action N-samples was generated from a Hammersley low-discrepancy sequence (Hammersley, 1960). For the given UA framing, the following stationary operation states are possible for a given action $\mathbf{a}$: i) a "stable" AD operation where methane production

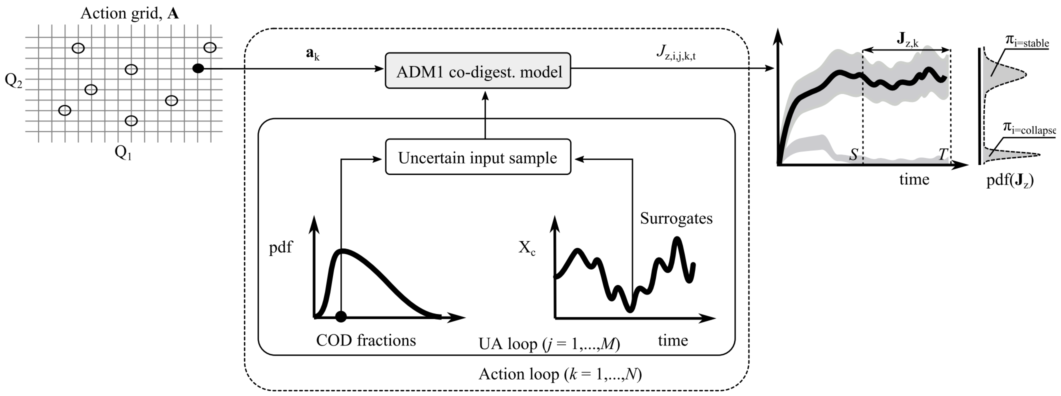


Figure 5.3.1: Uncertainty outcomes of the criterium $\mathbf{J}_z$ (PDF) for a given action $\mathbf{a}_k$ under input uncertainty.

is possible (i. e. $E = 1$ and index $i = \{stable\}$), ii) a "collapsed" AD operation where the system's buffer is broken and acidification occurs (i. e. $E = 1$ and $i = \{collapse\}$) and iii) an "unstable" AD operation where only for some input values the AD operation methane production is possible (i. e. $E = 2$ and $i = \{stable,\ collapse\}$). This means that at most two steady states are possible ($E_{max} = 2$): a "stable" AD is associated with methanogenetic bacteria retention, while an AD "collapse" is associated with its washout, caused by inappropriate operation conditions.

After the UA framing, the treatment plant's objectives and its admissible decision actions were defined. We propose a multi-criteria evaluation methodology under input the uncertainty and multiplicity, as described in the following steps:

Step 1. Take the k-th action $\mathbf{a}_k$ ($k = 1, ..., N$) from a designed action grid $\mathbf{A}$ (see Figure 5.3.1).

Step 2. Generate M uncertain input samples.

Step 3. For the given action $\mathbf{a}_k$ and for each input uncertainty sample, simulate the model over $S+T$ number of days. Discard the first S-days as start-up period.

Step 4. Repeat *Step 3* for $j = 1,\ldots,M$ (UA loop). Store the set of outcomes $\mathbf{J}_k$ relative to the stationary period (see Figure 5.3.1). Approximate the $M \times (T - S)$ uncertain set of observations $\mathbf{J}_k$ with the GMM probabilistic model, $g(\mathbf{J}_k) = g_k$. In particular, experiment with a range of E values ($E = 2,\ldots,E_{max}$) in order to find the most appropriate GMM model for $\mathbf{J}_k$. If for every E value $s < s^*$, fix E to 1 and store the relative GMM.

Step 5. Compute the expected utility multivariate normal distribution

$$u_k = u(\mathbf{a}_k) = Norm\left(\mu^U = \sum_{i=1}^{E} \pi_i \boldsymbol{\mu}_i,\ \Sigma^U = \sum_{i=1}^{E} \pi_i^{'} \boldsymbol{\Sigma}_i \pi_i\right)_k . \tag{5.3.6}$$

Step 6. Repeat *Step 1* to *Step 5* for all N samples (action loop in Figure 5.3.1) from the actions grid $\mathbf{A}$.

The k-th expected utility multivariate normal distribution u_k, defined in Eq. (5.3.6), is simply a “center of mass” distribution of its k-th multi-modal GMM distribution. In other words, given the i-th (i = 1, ..., E) sub-group, multi-criteria, stochastic variable

The k-th expected utility multivariate normal distribution u_k, defined in Eq. (5.3.6), is simply a “center of mass” distribution of its k-th multi-modal GMM distribution. In other words, given the i-th (i = 1,. . . ,E) sub-group, multi-criteria stochastic variable
$\mathbf{J}_i(\mathbf{a}_k)$,

$$\mathbf{J}_i\left(\mathbf{a}_k\right) \sim g_i\left(\mathbf{J}\left(\mathbf{a}_k\right)|\boldsymbol{\theta}_i\right), \tag{5.3.7}$$

the expected utility principle (Parmigiani and Inoue, 2009) consists of choosing the action $\mathbf{a}$ that maximizes the expected value of

$$\mathbf{U}_k = \frac{\sum_{i=1}^{E} \pi_{i,k}\mathbf{J}_{i,k}}{\sum_{i=1}^{E} \pi_{i,k}} \sim u\left(\mathbf{a}_k\right). \tag{5.3.8}$$

In our case, $\mathbf{U}$ is a stochastic Z-dimensional multivariate outcome and thus we are more interested finding Pareto-efficient actions based on the mean vector $\boldsymbol{\mu}^U = [\boldsymbol{\mu}^U_{1(CH4)}\ \boldsymbol{\mu}^U_{2(CODrem)}]$. The expected utility mean $\boldsymbol{\mu}^U$ can be considered as a real-valued summary of the worthiness of the outcomes that *may* result from it, while the corresponding standard deviation $\boldsymbol{\sigma}^U$ expresses the uncertainty about this summary. As the Neumann-Morgenstern utility theory requires (Parmigiani and Inoue, 2009), the GMM mixing proportions $\pi_i(\mathbf{a}_k)$ are considered fixed since they are regarded as a description of a well-understood chance mechanism, i. e. the UA framing. Note that the relative choice of s^* depends on the degree of “dissimilarity” that we expect between the clusters of steady-state points: in our case we chosen a relatively high $s^* = 0.8$ because “collapse” and “stability” for an AD-system are very “dissimilar”. Thus, fixing s^* is performed with regard to common sense and trial-and-error procedure.

Note that for a given action $\mathbf{a}_k$ the GMM models would result in an inappropriate description of the overall outcomes if the number of uncertain input samples (i. e. the value of M) is too low. In order to estimate an adequate number of M, we propose a simple "greedy" method:

Step 1. Compute the $\mathrm{GMM}_{\mathrm{old}}$ model with M_{old} samples for a known unstable or stable AD case (action $\mathbf{a}^*$).

Step 2. Compute the $\mathrm{GMM}_{\mathrm{new}}$ model with action $\mathbf{a}^*$ using $M_{new} = M_{old} + \Delta$ samples.

Step 3. If the difference between $\boldsymbol{\theta}_{i,new} = (\mu_{i,new}, \boldsymbol{\Sigma}_{i,new})$ and $\boldsymbol{\theta}_{i,old}$ is significant, set $M_{old} = M_{new}$ and repeat ii) and iii).

Step 4. Otherwise, compute GMM_{new} and GMM_{old} on a different randomly chosen action $\mathbf{a}^{**}$ (unstable or stable AD case should result) and check if the test of step iii) is still satisfied. If it is, then M_{new} is a valid number of MC samples, otherwise, go to step ii).

In our case study, the multi-criteria space was two dimensional ($Z = 2$), and the number of action samples was limited to $N = 80$ because of computational complexity issues. We found that for $M = 200$ samples (number of COD fractions and profiles) the above "greedy" algorithm was satisfied (initial try with $M_{old} = 50$ and $\Delta = 10$). The start-up simulation period was 640 days ($S = 640$), while the steady-state period was 960 days ($T = 1580$). The sampling time was fixed to 4 days because the auto-correlation information was not required to build the GMM models. This choice intuitively suggests that sufficient MC samples ($M \times (T - S)/4 = 47{,}000$) were used to calibrate the GMM model for a given action $\mathbf{a}_k$. The initial conditions of the simulated bio-gas plant were kept constant. Their values were found at the steady-state for a pig-manure feed fixed to a nominal value of 300 m^3/d in order to simulate the scenario where a plant operator switch from digestion of pig-manure alone to co-digestion. It would be easy to model the initial biomass concentration uncertainty since its relative probability distribution can be directly obtained from the nominal, MC-scenario. However, since in our UA-framing the growth/decay yields are assumed well known the resultant uncertainty of the initial biomass would be low. Thus, the additional biomass-uncertainty would just slightly change the outcome uncertainty distribution.

5.3.6 Gaussian Process Regression (GPR)

The computation of only $N = 80$ action samples (a total of $N \times M = 16{,}000$ model runs) required approximately 14 hours of simulation on a Core2 Duo PC with 2 GB RAM, under the Windows XP operating system. It is obvious, that if a denser mapping of $\boldsymbol{\mu}^U$ and $\boldsymbol{\sigma}^U$ is desired, an alternative to the "brute-force" approach is necessary. One approach is to use meta-models (emulators) as a replacement for the original simulation model. The Gaussian-process regression (O'Hagan, 1978) emulator is an interesting choice for non-linear interpolation problems because of its relatively simple structure definition and, due to its probabilistic nature, its automatic estimation of the prediction variance. This means that a complex nonlinear simulation model containing stochastic elements is replaced with emulator that is a simpler, but gives equivalent probabilistic and prediction behavior.

Here, modeling with Gaussian process regression (GPR) is presented only in brief; for a more detailed explanation see, e. g., Rasmussen and Williams (2006). Assume that the dependent variables y have a functional relationship of the form

$$y_i = f(\mathbf{x}_i), \tag{5.3.9}$$

where $\mathbf{x}$ denotes an input vector (covariates) of dimension D. The Gaussian process is a random function, fully described by its mean and variance. Gaussian processes can be viewed as a collection of random variables y_i with a joint multivariate Gaussian distribution $\mathbf{y} \sim Norm(\mathbf{0}, \mathbf{K})$. The covariance between the values of the function y_i and y_j is expressed by the elements $K_{i,j}$ of the covariance matrix $\mathbf{K}$ as $\mathrm{cov}(y_i,\ y_j) = K_{i,j} = C(\mathbf{x}_i,\ \mathbf{x}_j)$. Any function $C(\mathbf{x}_i,\ \mathbf{x}_j)$ can be a covariance function, providing it generates a non-negative definitive covariance matrix $\mathbf{K}$. One of the many choices for the covariance function $C(\,\cdot\,,\cdot\,)$ that we found suitable in our case is a rational quadratic (RQ) covariance function

$$C\left(\mathbf{x}_i, \mathbf{x}_j\right) = C_{RQ}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \sigma_s^2 \left\{1 + \frac{\left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{R}\left(\mathbf{x}_i - \mathbf{x}_j\right)}{2\gamma}\right\}^{-\gamma}, \tag{5.3.10}$$

where $\mathbf{R}$ is a diagonal matrix of D roughness parameters r_d (d = 1, ..., D), σ_s^2 is the estimate of the vertical scale of variation and γ is the shape parameter determining the diffuseness of the length-scales. The covariance matrix $\mathbf{K}$ depends on the selected covariance-function parameter vector $\boldsymbol{\varphi} = [r_1\ ...\ r_D\ \sigma_s^2\ \gamma]$, which is estimated using the training data ($\mathbf{X}$, $\mathbf{y}$) where $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2 \ldots \mathbf{x}_N]$ is the $D \times N$ *design matrix* and $\mathbf{y} = [y_1\ y_2\ ...\ y_N]$. A plausible estimate of the GPR parameter vector $\boldsymbol{\varphi}$ can be obtained by minimizing the log-marginal likelihood

$$\log p\left(\mathbf{y}|X\right) = -\frac{1}{2}\mathbf{y}^T\mathbf{K}^{-1}\mathbf{y} - \frac{1}{2}\log \mid \mathbf{K} \mid - \frac{N}{2}\log\left(2\pi\right). \tag{5.3.11}$$

The minimization of the log-marginal likelihood is computationally demanding since the inverse of the data covariance matrix $\mathbf{K}$ ($N \times N$) has to be calculated at every iteration. On the other hand, there are only $D + 3$ GPR parameters to be estimated, which means that the "curse of dimensionality" is less problematic than for other black-box statistical methods.

Once the Gaussian process regression (GPR) parameters are known, it is, due to probabilistic nature of emulator, possible to calculate a normal predictive distribution $y^* \mid (\mathbf{X}, \mathbf{y}), \mathbf{x}^*$ for a given new prediction input $\mathbf{x}^*$ as (Rasmussen and Williams, 2006)

$$\mu_{GP}(\mathbf{x}^*) = \mathbf{k}(\mathbf{x}^*)\mathbf{K}^{-1}\mathbf{y} \tag{5.3.12}$$

$$\sigma^2_{GP} = \kappa(\mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*)\mathbf{K}^{T-1}\mathbf{k}(\mathbf{x}^*) \tag{5.3.13}$$

where $\mathbf{k}(\mathbf{x}^*) = [C(\mathbf{x}_1, \mathbf{x}^*)\ C(\mathbf{x}_2, \mathbf{x}^*)\ \ldots\ C(\mathbf{x}_N, \mathbf{x}^*)]$ is the $N \times 1$ vector of the covariance between the training and the prediction cases, and $\kappa(\mathbf{x}^*) = C(\mathbf{x}^*, \mathbf{x}^*)$ is the auto-covariance of the prediction input. The prediction variance can be considered as a measure of confidence in model predictions, which is based on training data density and location. The utility of GPR is that of standard regression except that it does not require a structure resembling the system to be modeled but relying solely on the emulation of input-output behavior.

The procedure to replace a complex nonlinear AD model with a numerically more inexpensive, but empirically equivalent black-box GPR model was employed in our case as follows:

Step 1. A representative set of input data, $\mathbf{X}$, and target data, y, that cover all situations of interest was generated using the simulation model at hand. Inputs were the actions $\mathbf{X} = \{\mathbf{a}_k : k = 1, \ldots, N\}$ sampled from the action grid $\mathbf{A}$. Since for the considered RQ covariance function only a multiple-input, single-output emulator can be build, we constructed one GPR model for each output of interest: $\boldsymbol{\mu}^U_{CH4}$, $\boldsymbol{\mu}^U_{CODrem}$, $\boldsymbol{\sigma}^U_{CH4}$, $\boldsymbol{\sigma}^U_{CODrem}$ and $\boldsymbol{\pi}_{collapse}$. In this way, for example, it is possible to obtain for any action $\mathbf{a}^*$ of $\mathbf{A}$ the relative value $\mu^U_{GP,CH4}$.

Step 2. GPR model parameters for the selected covariance function, Eq. (5.3.10), were optimized so that the GPR response is as close to the target one according to the cost, Eq. (5.3.11).

Step 3. GPR model was validated with the comparison of responses using data that were not used for identification of GPR model (20% of the training data set) to confirm its usefulness for further analyses.

Step 4. The original complex simulation model was replaced with GPR model for the following analyses.

5.3.7 Pareto Front of the Expected Utility

The Pareto front of $\boldsymbol{\mu}^U$ was computed within a "non-dominated sorting" of 10,000 GPR model samples $\boldsymbol{\mu}^U_{GP} = [\boldsymbol{\mu}^U_{GP,CH4}\ \boldsymbol{\mu}^U_{GP,CODrem}]$. The goodness of the fit for the relative $\boldsymbol{\mu}^U$ was expressed in terms of the GPR prediction's standard deviation $\boldsymbol{\sigma}^U_{GP}$. In order to find the Pareto front over higher dimensional spaces the "non-dominated sorting" approach can be substituted, for example, with some multi-objective genetic algorithm in order to reduce the number of GPR model evaluations. In any case, without a fast and reliable system emulator, any iterative optimization algorithm would still require an unreasonable amount of time to converge to the Pareto front.

5.4 Results

A multi-criteria evaluation of the performance of the AD plant is essentially based on Eq. (5.3.6), where the expected utility multivariate normal distribution u_k is evaluated for each k-th decision operation action. Thus, the first result represented in Figure 5.4.1 is the expected utility distributions u_k for each k-th action ($k = 1,..., N$). The gray ellipses are the 95% percentile contours of the respective u_k distributions, while the black dots represent their midpoints ($\boldsymbol{\mu}^U_k$). The continuous segment from point A to B is the GPR-approximated Pareto front, while the discontinuous segments are the upper and lower error bounds estimated from $2\boldsymbol{\sigma}^U_{GP}$ (95% prediction interval). Note that the GPR approximation assumes noise-free training data. We highlighted two interesting utility-distribution centroids: point A represents the scenario where the removal of the COD is maximized, while point B represents the scenario where the methane production is maximized. Point A is interesting from a practical point of view if a decision maker focus only on environmental issues, e. g., when dealing with ecologically sensitive rural areas. In our case, approximately 15% of the COD removal capacity is lost if the methane production is prioritized; on the other hand, methane production is increased by four times. Note how the prediction interval of the estimated Pareto front is wide in areas where the training input space is sparse (e. g., near point A), while it is narrow in areas where training data are available (e. g., near point B). An operator is free to choose an efficient policy located on the segment between those two points, depending on its

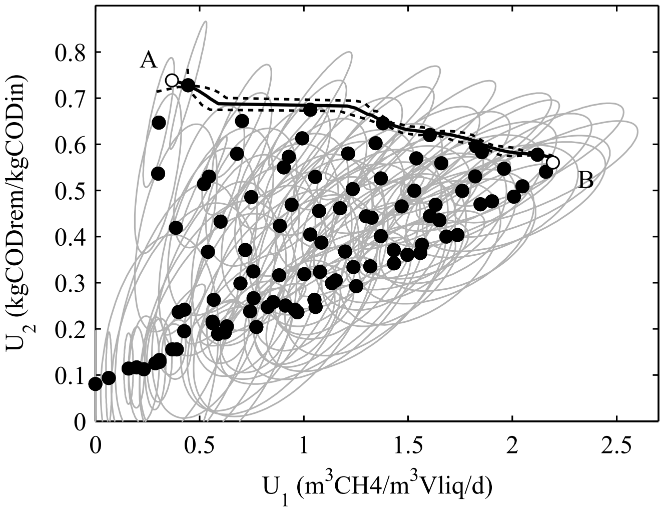


Figure 5.4.1: Expected utility u_k 95% percentile contours (gray ellipses) and the corresponding midpoints (dark dots). The continuous segment from point A to B is the GPR-approximated Pareto front $\boldsymbol{\mu}_{GP}^{U}$ in the multi-criteria space, while the discontinuous segments are the $2\boldsymbol{\sigma}_{GP}^{U}$ upper and lower prediction intervals for the GPR-approximated Pareto front.

objectives. If needed, one could always perform additional UA simulations in order to refine the Pareto-front estimation where the confidence region is too wide.

The second important result is the information about the process reliability for a given control action **a**. The process is unreliable if the AD reactor collapses. The probability of an AD collapse for a given action **a** is provided by the GPR-approximated mixing proportion $\boldsymbol{\pi}_{GP,collapse}$. This probability is represented in Figure 5.4.2 as a "process reliability map". The continuous segment from point A to B is the set of actions that are Pareto efficient. Note that those actions are all 100% reliable, but are bordering the region where 100% reliability is not guaranteed any more. We observe that if methane production is to be maximized by increasing the co-substrate, rich-COD, inflow load Q_2 (moving from point A to B), then the addition of the buffering solution Q_1 (pig manure) is essential if a reliable AD operation is desired. Point B is a crossway (Figure 5.4.1 and 5.4.2). In fact the system's 100% reliability and methane productivity cannot be satisfied at the same time just by adding the pig-manure substrate. If we maintain Q_1 as constant at the nominal value of point B and increase the co-substrate COD loading rate Q_2, the biogas yield *may* increase, but the reliability of the process will decrease. If Q_1 is increased in an attempt to make the plant operation more robust,

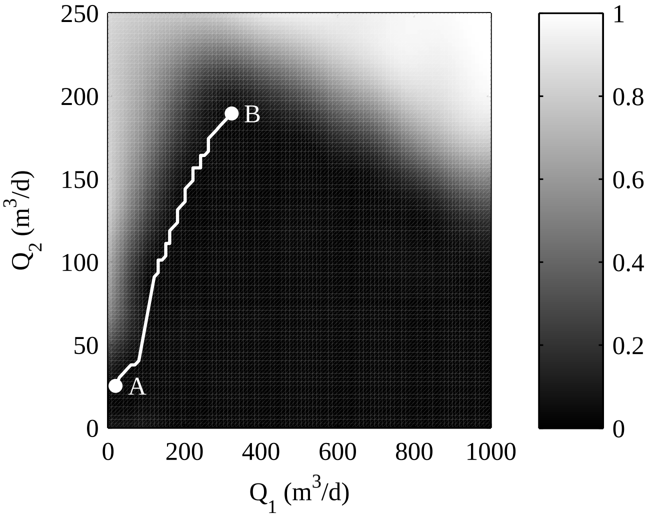


Figure 5.4.2: Process-reliability map where $\boldsymbol{\pi}_{GP,collapse}$ is the probability of an AD plant collapse under a given control action $\mathbf{a} = [Q_1\ Q_2]$. The continuous segment from point A to B is the GPR-approximated Pareto front $\boldsymbol{\mu}^U_{GP}$ in the action space $\mathbf{A}$.

biomass washout occurs: the plant's collapse probability increases within a decrease of the expected methane productivity. If one tries to force the system to produce more methane than it produces at point B, a "slide-down" effect of the expected criteria midpoints is observed in Figure 5.4.1.

Finally, the GPR mapping between the actions and the expected utility means $\boldsymbol{\mu}^U_{GP,z}$ and the standard deviations $\boldsymbol{\sigma}^U_{GP,z}$ is represented in Figure 5.4.3. Actions that are Pareto efficient are also represented. Under the methane production maximum policy (point B) the expected methane yield is 2.3 ±0.50 m^3CH_4/m^3Vliq/d. On the other hand, if the COD removal is maximized then the expected methane yield is 0.4 ±0.08 m^3CH_4/m^3Vliq/d. The standard deviation $\boldsymbol{\sigma}^U_{GP,CH4}$ related to the expected utility of methane is not constant. In particular, if only the Pareto-efficient actions are considered, the standard deviation increases by 0.10 m^3CH_4/m^3Vliq/d for a unit increase of methane production. The relationship is linear. This result can be explained by the fact that the uncertainty associated with the inert COD fraction scales with the average inflow COD loading. On the other hand, if we consider the expected utility of the COD removal efficiency (see 5.4.3, lower right), we observe that its standard deviation $\boldsymbol{\sigma}^U_{GP,CODrem}$ is more or less constant at 0.06 kgCOD/kgCOD (only Pareto-efficient actions considered). The value of $\boldsymbol{\sigma}^U_{GP,CODrem}$ is mainly due to the uncertainty in the COD inert fraction

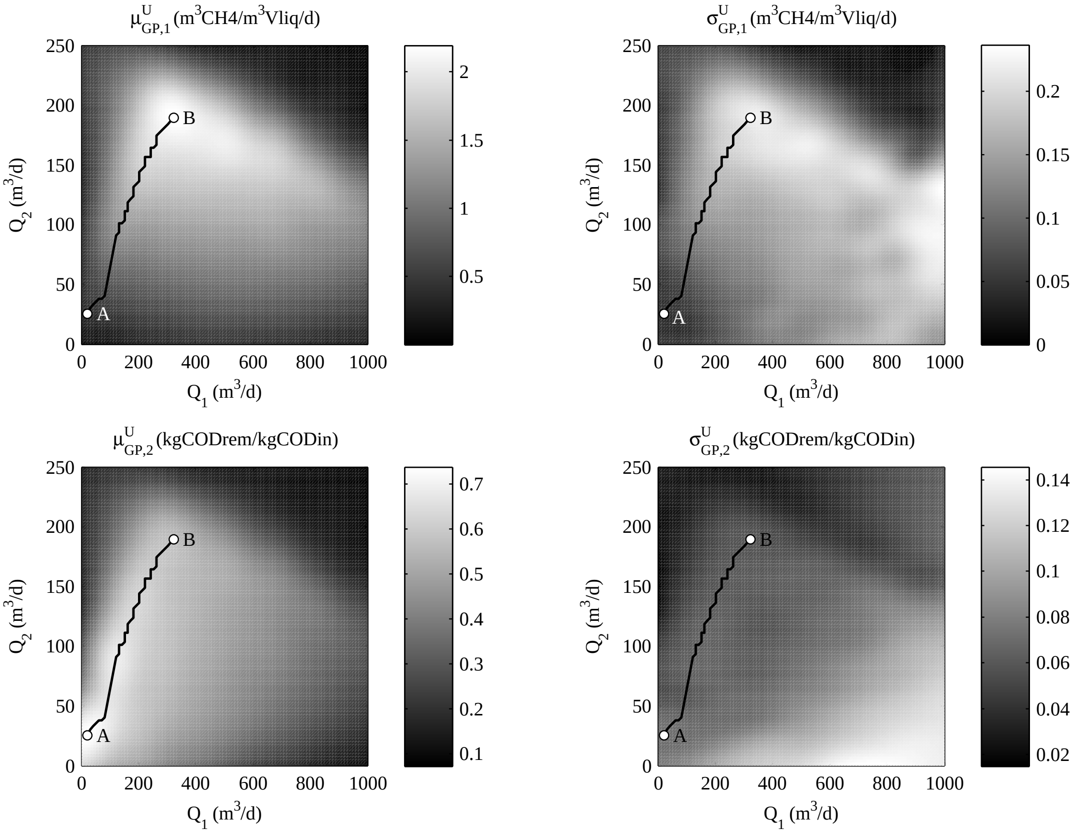


Figure 5.4.3: Expected utility means $\boldsymbol{\mu}^{U}_{GP,z}$ (left) and standard deviations $\boldsymbol{\sigma}^{U}_{GP,z}$ (right) for the respective criteria (methane production (up) and COD removal (down)). The continuous segment from point A to B represents actions that are Pareto efficient. The methane mean $\boldsymbol{\mu}^{U}_{GP,CH4}$ and its standard deviation $\boldsymbol{\sigma}^{U}_{GP,CH4}$ are linearly dependent. The grayscale colors indicate the intensity of the respective measures.

parameters. The best expected COD removal efficiencies $\boldsymbol{\mu}^U_{GP,CODrem}$ is approximately 73% and, as expected, it is obtained for a high hydraulic retention time (HRT) at low flow rates (see point A in Figure 5.4.3). In fact, the microorganism's activity is still able to process the majority of the inflow biodegradable COD. However, if the inflow COD load is increased, the HRT decrease: $\boldsymbol{\mu}^U_{GP,CODrem}$ decreases at most by 15% because the inflow COD increase is still compensated by an increase in the degraders activity (increase in biomass population).

5.5 Discussion

It is important to point out that any result from a UA is closely related to its framing (Sin et al., 2009): if we are happy with the partial elicitation of uncertainty around the prediction in question, then we should expect reasonable results. However, the contrary is always possible. In our case, only inflow COD fractions were assumed to be uncertain. Other important uncertain inputs may be considered: hydrolysis rates, biomass yields, etc. In this context, global sensitivity analysis is a valuable tool in order to decide which model-inputs (or factors) influence most a given model-output (Sin et al., 2011). The advantage of UA is that assumptions are explicitly stated and thus, they can be changed at will by different analysts. The task of a treatment-plant operator is to critically assess the results relative to the given UA framing.

We used a linear affine transformation in order to reduce the multi-modal, multi-criteria PDF to a uni-modal one. The resulting PDF was the expected utility distribution, which is a kind of "center of mass" or "centroid" of the original PDF. There are a number of other possible transformations; however, the linear one is easy to interpret and commonly applied (e. g., the defuzzification process in fuzzy logic).

The estimation of the expected utility Pareto front from $\boldsymbol{\mu}^U_{GP}$ makes it possible for a WWTP operator to select efficient actions. The computation of these efficient actions is fast because we introduced an approximation to the initial problem. It is thus important to evaluate how good the approximation is. In our case, the GPR emulator was able to give an estimate of this approximation within its predictive standard deviation $\boldsymbol{\sigma}^U_{GP}$. However, this estimation depends on the particular covariance function $C(\mathbf{x}_i, \mathbf{x}_j)$ chosen. The covariance function $C(\mathbf{x}_i, \mathbf{x}_j)$ represents our a-priori knowledge about the shape of the function $f(\mathbf{x})$ in Eq. (5.3.9) to be emulated. Because a rational quadratic (RQ) covariance function was selected, we implicitly made the assumption that $f(\mathbf{x})$ is infinitely mean-square differentiable (i. e. smooth). Thus, the prediction variance estimation $\boldsymbol{\sigma}^U_{GP}$, and the GPR emulator itself, depend not only on the number of available training data, but also on how reasonable our assumptions are about the shape of $f(\mathbf{x})$.

The results found here agree with those found by Shen et al. (2007) for a two-population anaerobic digestion model: the addition of an alkali solution to the reactor improves the robustness by enlarging the attractive domain of the product-formation equilibria. In our case the buffering agent was the pig-manure substrate. If the buffering agent was increased while maintaining a constant co-substrate feed Q_2, the stability region (the dark area in Figure 5.4.2) of the AD process reliability map was enlarged. We similarly found that increasing the inflow co-substrate load leads to high methane yields, but reduces the robustness of the system if no buffering solution compensation is provided. This compensation cannot last indefinitely because of biomass washout.

Analyzing the expected utility Pareto front the conflict between the methane production (economic objective) and the COD removal capacity (environmental objective) of the plant is evident. If a plant operator would like to maximize methane production (point A in Figure 5.4.1), then the COD removal efficiency would be inevitably compromised because he/she would need to provide high pig-manure substrate loads in order to maintain a stable process. A low HRT implies that only the readily available bio-degradable organic material is converted into methane, while other, slowly bio-degradable, fractions are passed through the system “untouched”.

In practice, if a plant operator (or investor) is interested in the economic feasibility of a current or future bio-process plant, the dispersion measure of the product formation can give a sense of the risks/opportunities associated with the average expected monetary benefit under the chosen operation scenario. In our case, we found a linear relationship between the mean methane production $\boldsymbol{\mu}^{U}_{GP,CH4}$ and its standard deviation $\boldsymbol{\sigma}^{U}_{CP,CH4}$ (Figure 5.4.3, upper row). As in modern portfolio theory (Markowitz, 1952), there is still an “investment” risk associated with the standard deviation of the return that should be accounted for.

The main advantage of the proposed UA methodology is that it can be applied to systems that present multiplicity and that the MC computational complexity is reduced by an appropriate approximation, the goodness of which can be evaluated. However, the GPR model used for the approximation has its own computational limits: it can be applied to problems with a moderate number of action dimensions ($D < 30$) and training samples ($N < 1{,}000$) at the present level of computer technology and interpreted code.

5.6 Conclusions

We have presented a method to conduct multi-criteria evaluations under uncertainty and systems multiplicity. It was applied to an anaerobic, co-digestion version of ADM1 in order to estimate the set of operation actions that are Pareto efficient. The actions were performed over the dilution rates of the substrates. First, we show how a multi-modal, multi-criteria distribution can be reduced within the utility theory into a standard uni-modal, multi-criteria, analysis problem. Second, in order to estimate the expected Pareto front, we apply a novel statistical approximation technique known as Gaussian process regression. Because a Gaussian process is a random function, fully described by its mean and variance, the goodness of the expected Pareto-front approximation can be assessed. Furthermore, the computational complexity is considerably reduced.

The results show that plant reliability is strongly dependent on the proportions of the buffering-solution substrate (pig manure) and the co-substrate, rich-COD (beet energy crop) inflow loads. The estimated plant-reliability map showed how an increase in the pig-manure substrate load promotes an enlargement of the stability region of the plant's operation. In this way, co-substrate loads can be increased in order to achieve higher methane productivity. However, this stability enlargement is not able to sustain indefinite increases of co-substrate loads since methanogenic bacteria washout occurs. Thus, an inherent trade-off between robustness and productivity is observed. The estimated Pareto front shows the degree of conflict between the economic objective of methane production and the environmental objective of COD removal. We found that even if Pareto-efficient actions are taken, the standard deviation of methane production is linearly dependent on its mean value. This implies that policies aiming to increase methane yields will have inherently high production uncertainties. In this case, a mean-variance compromise should be looked for.

Finally, it should be emphasized that the proposed multi-criteria evaluation approach covers only the case where plant-operation optimization is performed in a static way. Actions are fixed over time and a static regression model is built in order to emulate the plant outcomes under uncertainty. However, a possible strategy for the future would be to build a dynamic Gaussian process model in order to perform a dynamic optimization of the plant under uncertainty.

Acknowledgments

This research was partially supported by the Spanish Ministry of Science and Innovation (Project CTM2010-18212) and the Slovenian Research Agency (project L2-2157). The authors wish to thank Dr. A. Magrí Aloy for its constructive comments.

Bibliography

Bastone, D. J., Keller, J., Angelidaki, I., Kalyuzhnyi, S. V., Pavlostathis, S. G., Rozzi, A., Sanders, W. T. M., Siegrist, H., Vavilin, V. A., 2002. Anaerobic digestion model no.1 (ADM1). Tech. rep., IWA Publishing , UK.

Benedetti, L., Baets, B. D., Nopens, I., Vanrolleghem, P., 2010. Multi-criteria analysis of wastewater treatment plant design and control scenarios under uncertainty. Environmental Modelling & Software 25 (5), 616–621.

Chen, Y., Cheng, J. J., Creamer, K. S., 2008. Inhibition of anaerobic digestion process: a review. Bioresource Technology 99 (10), 4044–4064.

Elnashaie, S. S. E. H., Uhlig, F., Affane, C., 2007. Numerical Techniques for Chemical and Biological Engineers Using MATLAB: A Simple Bifurcation Approach. Springer, USA.

Flores-Alsina, X., Rodríguez-Roda, I., Sin, G., Gernaey, K. V., 2008. Multi-criteria evaluation of wastewater treatment plant control strategies under uncertainty. Water Research 42 (17), 4485–4497.

Galí, A., Benabdallah, T., Astals, S., Mata-Alvarez, J., 2009. Modified version of ADM1 model for agro-waste application. Bioresource Technology 100 (11), 2783–2790.

Gernaey, K. V., Flores-Alsina, X., Rosen, C., Benedetti, L., Jeppsson, U., 2011. Dynamic influent pollutant disturbance scenario generation using a phenomenological modelling approach. Environmental Modelling & Software 26 (11), 1255–1267.

Hammersley, J. M., 1960. Monte carlo methods for solving multivariable problems. Annals of the New York Academy of Sciences 86 (3), 844–874.

Henze, M., Gujer, W., Mino, T., van Loosdrecht, M., 2000. Activated sludge models ASM1, ASM2, ASM2d and ASM3. Tech. rep., IWA Publishing. London, UK.

Markowitz, H. M., 1952. Portfolio selection. Journal of Financial 7 (1), 77–91.

McLachlan, G., Peel, D., 2000. Finite Mixture Models. John Wiley and Sons, USA.

Oakley, J. E., O'Hagan, A., 2004. Probabilistic sensitivity analysis of complex models: a Bayesian approach. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66 (3), 751–769.

O'Hagan, A., 1978. Curve fitting and optimal design for predictions. Journal of the Royal Statistical Society 40 (1), 1–42.

O'Hagan, A., Oakley, J. E., 2004. Probability is perfect, but we can't elicit it perfectly. Reliability Engineering & System Safety 85 (1-3), 239–248.

Parmigiani, G., Inoue, L. Y. T., 2009. Decision Theory: Principles and Approaches. John Wiley and Sons, USA.

Rasmussen, C. E., Williams, C. K. I., 2006. Gaussian Processes for machine learning. The MIT Press, Cambridge, USA.

Rosen, C., Jeppsson, U., 2006. Aspects on ADM1 implementation within the BSM2 framework. Tech. rep., Department of Industrial Electrical Engineering and Automation, Lund University, Sweden.

Rosen, C., Vrečko, D., Gernaey, K. V., Pons, M.-N., Jeppsson, U., 2006. Implementing ADM1 for plant-wide benchmark simulations in MATLAM/Simulink. Water Science and Technology 54 (4), 11–19.

Rousseeuw, P. J., 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Computational and Applied Mathematics 20, 53–65.

Sbarciog, M., Loccufier, M., Noldus, E., 2010. Determination of appropriate operating strategies for anaerobic digestion systems. Biochemical Engineering Journal 51 (3), 180–188.

Schreiber, T., Schmitz, A., 2000. Surrogate time series. Physica D: Nonlinear Phenomena 142 (3-4), 346–382.

Shen, S., Premier, G., Guwy, A., Dinsdale, R., 2007. Bifurcation and stability analysis of an anaerobic digestion model. Nonlinear Dynamics 48 (4), 391–408.

Sin, G., Gernaey, K. V., Neumann, M. B., van Loosdrech, M., Gujer, W., 2011. Global sensitivity analysis in wastewater treatment plant model applications: Prioritizing sources of uncertainty. Water Research 45 (2), 639–651.

Sin, G., Gernaey, K. V., Neumann, M. B., van Loosdrecht, M., Gujer, W., 2009. Uncertainty analysis in WWTP model applications: A critical discussion using an example from design. Water Research 43 (11), 2894–2906.

Volcke, E., Sbarciog, M., Noldus, E., Baets, B. D., Loccufier, M., 2010. Steady state multiplicity of two-step biological conversion systems with general kinetics. Mathematical Biosciences 228 (2), 160–170.

Wang, R., Diwekar, U., Gragoire Padro, C. E. G., 2004. Efficient sampling techniques for uncertainties in risk analysis. Environmental Progress 23 (2), 141–157.

6 General Conclusions & Outlook

6.1 General Conclusions

In this thesis, the main focus was on engineering modeling under uncertainty. Engineering modeling not only contemplates process-modeling, but also modeling decisions based on an explicit model of a process. Within this framework, the main contributions of the work at hand can be summarized as follows:

- A simulation environment called “virtual plant” (VP) was build in order to simplify the engineering process-modeling step. It was applied to anaerobic co-digestion and activated sludge modeling, but other biotechnological processes may be considered for modeling within this simulation environment.
- In ADM environment, new core dynamics of the long chain fatty acids (LCFA) inhibition process were proposed and tested and it was shown that saturated/unsaturated LCFA-degraders are determinant for the evolution of the system.
- In ASM environment, instructive comparison between two popular procedures for model-parameter inference were presented.
- A novel multi-criteria evaluation under model uncertainty in ADM environment was developed for biotechnological processes that present multiplicity of equilibrium.

6.1.1 Modeling Toolkit for Engineering

Even if C/C++, Java and Python rank high in the TIOBE software index, the population of users is mainly professional programmers and not engineers. Arguably, the most popular “programming language” used by engineers is still Excel. One reason is that engineers are not programmers, but modelers. In particular, they model decision processes. However, today complex engineering decision problems need powerful and flexible modeling environments and Excel was certainly not designed for this task. In engineering practice, simple models are generally favored over complex models because of sparse data, ease of interpretation and programming language functionality limitations. Overcoming the latter implies learning a new programming language or a dedicated software. This process is both expensive and time consuming, especially if previous programming skills cannot be reused to accelerate the learning process. For this reason, linking Excel to a powerful programming language may accelerate this transition process: the user may feel comfortable within an Excel interface, while gradually migrating to a more powerful simulation tool. A second reason for the popularity of Excel is its presence in almost any office. In this way, an exchange of models between engineers and their maintenance are promoted.

In this PhD work, a “virtual plant” toolkit was developed for engineering modeling and decision making (Chapter 2). Linking Excel to MATLAB/Simulink was a positive experience: it permitted non-programmers to enter in the iterative process of progressive refinement of biotechnology models and to actively contribute to process optimization.

6.1.2 Modeling in Biotechnology: the case of LCFA-inhibition Process

In the present work, we focused on models based on ordinary differential equations and proposed two LCFA-inhibition process models (i. e. LCFA-M1 and LCFA-M2 in Chapter 3). Both models were proven to reproduce the main trends of a LCFA-inhibited system operated in a wide range of experimental designs. However, a systematic mismatch between the proposed models and measurements (auto-correlated residuals) was still present. This was the main reason why parameter uncertainty estimation was not performed for these models. Instead, confidence over the estimated parameter values was based on a global sensitivity analysis. The main result was that both models evidenced that the distribution of the saturated/unsaturated degraders plays an important role on the system evolution under LCFA-inhibition.

Damage of the cell-functionality was modeled by a new state variable (LCFA-M2). This approach was superior to the usual non-competitive inhibition function approach (LCFA-M1) when modeling sever LCFA-inhibitory events. The main advantage of the “healthy-state” variable was that it accounted for the processes that produced a lag-time between the complete LCFA concentration depletion and the re-start of methane production. Understanding and correctly predicting this cell-functionality recovery lag-time would permit the development of optimal control strategies and soft-sensors for bio-reactors feed with lipid rich substrates.

6.1.3 Estimation of Parameter Uncertainty

When nonlinear models in parameters are used for modeling and their parameters need to be inferred, two popular inferential procedures are available: Bayesian and frequentist. However, if the parameter value uncertainty is represented as a random variable, under particular hypotheses, the frequentist result is only a linear approximation of the parameter-uncertainty. Although the use of Bayesian inference is commonly applied within different fields of knowledge, it can be considered still as a new approach in the wastewater community. This work compared the results from the Bayesian and the linear (frequentist) procedures under different inferential scenarios. It was found that because there are many situations where a priori it is not possible to assess suitability of the linear approximation, it may result advisable to use Bayesian inference instead.

Special focus was put on the comparison between frequentist and Bayesian results under different inferential scenarios. In particular, when some parameters are non-identifiable, a common practice in frequentist procedures is to fix those parameters at some nominal values. The estimation of uncertainty is then performed over the remaining, less problematic, parameters. From the Bayesian point of view, this assumption implies that perfect knowledge of the values of the non-identifiable parameters are available, which is a self-contradiction. The Bayesian procedure allows describing the degree of ignorance about the "true" value of the non-identifiable parameters in a more coherent way: if no a prior knowledge is available, then a non-informative (or weak) informative prior is used. Under this inferential scenario, it was empirically shown that the frequentist and Bayesian uncertainty estimations return quite different results. The many possible causes of this difference were not studied in this work.

Sensitivity analysis of the Bayesian results plays a crucial role in determining the robustness of the inference. The random errors that are included in the model make the relationship between measurements and predictors a "statistical" one. Thus, a statistical model for the random errors must be assumed in advance. This assumption is as subjective as the assumption over the functional structure that describes the deterministic dynamics of the bio-process under study. When the "true" statistical model is unknown, many distributional families may be "equally" good candidates to model random errors. Estimations of parameter uncertainty that result from concurrent statistical models must be compared to verify the sensitivity of the Bayesian solutions. We limited our sensitivity analysis to a qualitative comparison. However, Bayesian procedures offer a powerful framework for model selection within Bayesian factors: a future goal would be to apply this framework in model selection problems.

We conclude with the remark that the Bayesian procedure is just a choice between many other inferential procedures: its use is more or less justified when the relative assumptions are fulfilled.

6.1.4 Optimization under Uncertainty and its Reduction

We have presented a novel method to conduct multi-criteria evaluations under uncertainty and multiplicity. This method, like other Monte Carlo methods, explicitly transforms input uncertainty, suggesting a degree of objectivity of the computed results. When input uncertainty elicitation is based only on subjective expert's beliefs (i. e.

knowledge), results are necessarily subjective. Contrary, if expert's beliefs are considered objective and rational, then results should be considered objective too. Thus, uncertainty analysis framing is of central importance in providing useful results.

Under specified uncertainty analysis framing, the multi-criteria evaluation method was applied to an industrial anaerobic co-digestion biogas plant. The method was found useful to optimize the bio-process in environmental and/or economic terms over a long run operation period, while preserving operation reliability. This results indicates that further improvement and application of such multi-criteria evaluation method is potentially beneficial to other industrial biotechnological processes.

Besides accounting for uncertainty in decision making, reducing uncertainty is another important task (see the following Appendix A). Additional research, data collection or experimental designs for reducing parameter uncertainty are generally expensive. From an engineering point of view, further information collection is valuable only if it reduces the likelihood of making the wrong decision. Thus, prioritization of uncertainty reduction should be considered in the context of decision making. During the last decade, because of increased risks and scarce resources, the expected value of information analysis in complex health and financial economic models has become a very popular procedure. Extending its use to engineering decision making, monitoring or planning is an ambitious task for the future.

6.2 Outlook

Virtual Plant (VP)

The VP toolkit (Chapter 2) is at its early stage of development and thus, as any new software, it needs a considerable debugging and code-optimization effort. One of the many solution to speed-up its development could be to make the code available under a free- or open-source license.

LCFA-inhibition Process Modeling

The proposed models for the LCFA-inhibition process (Chapter 3) needs to be falsified by the scientific community. Laboratory protocols for LCFA measurements in the solid and liquid phase should be improved in order to decrease experimental errors. A ranking of parameters that mostly contribute to the model-output variability is still missing and thus, a sensitivity analysis of parameters is needed. Moreover, an optimal experimental designs and a proper likelihood function that accounts for residual correlations should be built for a credible inference of parameters.

Parameter Inference

When data are sparse in relation to model-complexity and no prior knowledge is available over the possible values of the parameters, Bayesian formulation cannot *per-se* improve the parameter collinearity problem. However, Bayesian modeling may improve parameter non-identifiability when prior information and the imprecision of this information are entered in the inferential procedure. The frequentist procedure can be modified to handle collinearity by penalized likelihood (regularization), but the problem is that parameter-uncertainty estimation is *not* directly possible within this inferential framework. In the present work, the prior information was marginal in our analysis. However, the real strength of the Bayesian procedure is the possibility to deal with realistic situations in which informative prior knowledge can be taken into account and properly balanced with the experimental information.

In Chapter 4 was found that when modeling complex biological processes, “lumped” parameters that are assumed uncertain may vary depending upon where the data were collected. This indicated that parametric relations conditional on particular environmental factors should exist. A future challenge would be to specify a multilevel stochastic representation of such environmental variability within a hierarchical Bayesian modeling framework, trying to improve the extrapolation capacity of a particular bio-process model.

Optimal Decisions under Uncertainty

In our case study of a co-digestion biogas plant (Chapter 5), parameter uncertainty (i. e. COD fractions of the substrates) was modeled by expert’s knowledge base on experimental evidence. However, a more formal estimation of parameter uncertainty could have been done within the Bayesian inferential procedure because it combines prior expert’s knowledge and data. The advantage of using Bayesian inferential results as an input to uncertainty analysis for decision making is attractive especially for high-dimensional spaces with correlation parameter-structures. Moreover, some important uncertain parameters (i. e. $\mu_{max,ac}$ and $X_{ac}(0)$) where not included in the uncertainty analysis case study and thus, there are certainly margins in order to improve the uncertainty analysis framing.

The multi-criteria evaluation approach presented in this work has a practical limitation: actions are Pareto efficient only if hold over a long-run period. In other words, the optimal control strategy is of static type. Beside robustness, flexibility in the operation of a bio-reactor is a valuable attribute. A possible solution to increase the flexibility of the plant operation would be to design a model predictive control (MPC) schema that can work under model uncertainty. The main problem for the development of such MPC-schema is its computational intractability. One approach to reduce computational complexity is by approximation. Model prediction uncertainty could be approximated by Gaussian Process emulation of dynamic systems for multiple outputs. However, building such emulator machine may result as challenging as running the original uncertain model and thus, this option should be considered with care.

6.3 Concluding Remarks

The overall impression gained during this project can be best expressed by quoting Voltaire:

> *"Doubt is not a pleasant condition, but certainty is absurd."*

Engineering environmental modeling under uncertainty for decision-making is not warranted at the present state of technology and decision making culture. On one side, technological improvements of models relay on our understanding of complex environmental processes, on the willingness to admit our doubts about this knowledge, and to provide tools to characterize and quantify those doubts. On the other hand, decision-makers have their own doubts about the solutions proposed by engineers or scientists, which compete or are confronted with their own conceptual models for risk assessments. Integration of both systems of expertise is a necessary condition in order to achieve balance between the rational and the behavioural aspects of human action.

Appendix A. Value of Perfect Information Analysis

Pareto-efficient actions estimated in Chapter 5 are suboptimal because they should be robust enough in order to perform well for the many possible outcomes. Imagine that it is possible to perform additional experiments and obtain perfect information about the uncertain inputs. When uncertainty is removed, model-outcomes are certain and optimal actions are possible. Expected value of perfect information (EVPI) is the difference between the expected value conditional on perfect information (EV|PI) and the expected value under uncertainty (EV). Thus, value of perfect information (VPI) analysis addresses the question of how much is one willing to pay for perfect information before performing any action or decision.

In particular, assume we are faced with a set of possible actions $\mathbf{A} = \{\mathbf{a}_k : k = 1, \ldots, N\}$, which influence the state of a given system modeled by the explicit model $\mathbf{J}_k = f(\mathbf{a}_k, \mathbf{X})$, where $\mathbf{X}$ is the vector of uncertain inputs and $\mathbf{J}_k$ is the uncertain benefit (criteria). The overall EVPI scalar quantity for input $\mathbf{X}$ is estimated as

$$\begin{aligned} EVPI\,(\mathbf{X}) &= EV|PI - EV \\ &= E_{\mathbf{X}}\left\{\max_{\mathbf{a}}\mathbf{J}(\mathbf{a},\mathbf{X})\right\} - \max_{\mathbf{a}} E_{\mathbf{X}}\left\{\mathbf{J}\left(\mathbf{a},\mathbf{X}\right)\right\}. \end{aligned} \tag{6.3.1}$$

Consider our case study: the AD biogas plant of SAVA. The uncertain input set was defined as $\mathbf{X}$=$\{\{x_{ch},\ x_{pr},\ x_{li},\ x_{inert}\}_{1,2}, X_1,\ X_2\}$. Assume that $\mathbf{X}$ is partitioned in two subsets: $\mathbf{X}_p$ =$\{x_{ch},\ x_{pr},\ x_{li},\ x_{inert}\}_{1,2}$ is the subset of COD fractions relative to the pig manure substrate (index=1) and the beet energy crop co-substrate (index=2), while $\mathbf{X}_{\sim p} = \{X_1,\ X_2\}$ is the subset of the inflow COD concentrations for the relative substrates. Assume that it is possible to develop in the near future a new laboratory substrate characterization procedure. The request is that this new lab-procedure should provide perfect information over the values of the COD fractions. Contrary, assume that it is impossible to develop a procedure that can exactly predict the inflow COD concentration over a long period of time. In this scenario, what is the maximum quantity of resources provided by a rational policy maker for the development of the "exact" lab-procedure? The EVPI estimate is a point of reference for the policy maker: for example, by investing an EVPI amount of money into the "exact" lab-procedure project, the expected returns would be such that he did not gain nor lose any money by deciding to finance this project when compared to not financing the project. If the quantity of resources invested in the project is higher then the EVPI, the policy maker should expect to take a loss.

Since we are interested in the EVPI estimate for a subset of inputs $\mathbf{X}_p$ (COD fractions) the following partial EVPI is appropriate:

$$EVPI\left(\mathbf{X}_p\right) \quad = \quad E_{\mathbf{X}_p}\left\{\max_{\mathbf{a}} E_{\mathbf{X}_{\sim p}|\mathbf{X}_p}\left[\mathbf{J}(\mathbf{a},\mathbf{X})\right]\right\} - \max_{\mathbf{a}}\left\{E_{\mathbf{X}}\left[\mathbf{J}\left(\mathbf{a},\mathbf{X}\right)\right]\right\}. \quad (6.3.2)$$

Methane production efficiency, $\mathbf{J}_{1(CH4)}$, is easy to monetize and thus, it was considered for our VPI analysis. Note that an estimation of $E_{\mathbf{X}}\left[\mathbf{J}_{CH4}\left(\mathbf{a},\mathbf{X}\right)\right]$ is already available: it is the expected utility mean value for the methane production, $\boldsymbol{\mu}^U_{CH4}$ - Eq. (5.3.6).

The first term in the right-hand-side of Eq. (6.3.2) was estimated by using a simple two-level sampling algorithm (Tappenden et al., 2004; Oakley et al., 2010):

Step 1. Sample once from the group of inputs of interest ($\mathbf{X}_p$) and hold that inputs constant at their sampled value - *inner level* sample.

Step 2. Sample L values from the group of inputs *not of interest* ($\mathbf{X}_{\sim p}$) according to their prior uncertainty - *outer level* sample.

Step 3. For each of the L sampled inputs, obtain $\mathbf{J}_{CH4} = f(\mathbf{a},\mathbf{X}_p,\mathbf{X}_{\sim p,l})$ for $l = 1,\ldots,L$, and estimate

$$\begin{aligned} \max_{\mathbf{a}} E_{\mathbf{X}_{\sim p}|\mathbf{X}_p}\left[\mathbf{J}_{CH4}\right] \quad &\approx \quad \max_{\mathbf{a}} \frac{1}{L}\sum_{l=1}^{L}\frac{1}{T-S}\sum_{t=S}^{T}\mathbf{J}_{CH4,l,t} \\ &= \quad \max_{\mathbf{a}} \frac{1}{L}\sum_{l=1}^{L}\hat{\mathbf{J}}_{CH4,l} = \hat{m}_{CH4}\left(\mathbf{X}_p\right). \end{aligned}$$

Step 4. Repeat M times *Step 1-3* and compute the partial EV|PI as

$$E_{\mathbf{X}_p}\{\hat{m}_{CH4}(\mathbf{X}_p)\} \approx \frac{1}{M}\sum_{m=1}^{M}\hat{m}_{CH4}\left(\mathbf{X}_{p,m}\right).$$

The total number of model runs is $L \times M \times N$ since we have to account for the N number of action values sampled from the action grid $\mathbf{A}$. The expected steady state methane production $\hat{\mathbf{J}}_{CH4,l}$ is estimated with high precision because of the high number of samples (i. e. $T - S = 960$). Note that the following assumptions were necessary to make the above sampling algorithm feasible:

- Actions were assumed discrete, sampled from $\mathbf{A}$ as a Hammersley low-discrepancy sequence (Hammersley, 1960). Those action samples were fixed constant during the two-level sampling.

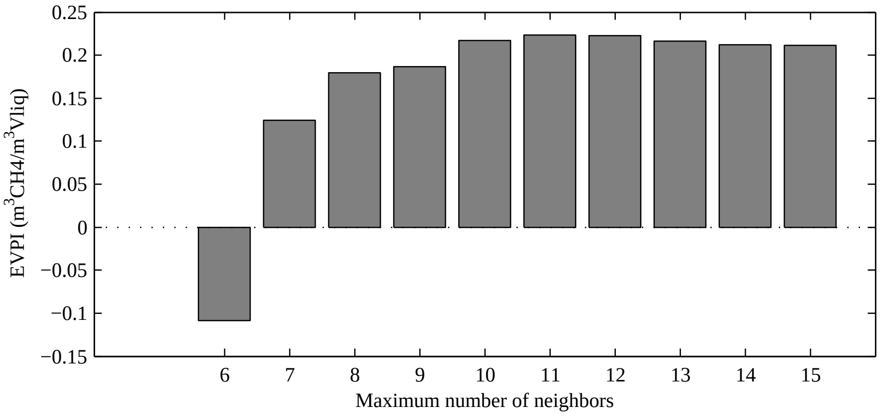


Figure 6.3.1: Sensitivity analysis of EVPI over the maximum number of neighbors parameter of the Lazy Learning smoother.

- Given $\hat{m}_{CH4}(\mathbf{X}_p)$, the number of parameter samples M is high enough to provide an unbiased estimate of $E_{\mathbf{X}_p}\{\hat{m}_{CH4}(\mathbf{X}_p)\}$ (i. e. EV|PI).
- The number of inner level samples L was fixed to one.

The last assumption is critical. In general, if L is very low the estimated EV|PI value result biased (Oakley et al., 2010), even if the number of outer level samples M is infinite. This problem was handled by assuming that $E_{\mathbf{X}_{\sim p}|\mathbf{X}_p}\left[\hat{\mathbf{J}}_{CH4}\right]$ can be approximated by a kernel smoother. The idea is to locally average $\hat{\mathbf{J}}_{CH4}$ in the neighborhood of $\mathbf{a}_k$. Assume that the variance induced by $\mathbf{X}_{\sim p}$ over $\hat{\mathbf{J}}_{CH4}$ is constant in the neighborhood of the relative nominal location $\mathbf{a}_k$. Lazy Learning local regression algorithm (Bontempi et al., 1997) was used as a local regression smoother and its “winner-takes-all” paradigm was applied: the best local approximation between a constant, a linear and a quadratic model is automatically selected. Lazy Learning was preferred to GPR because of its straightforward application to non-stationary noisy functions and unique parameter solution for a local model quarry.

The bandwidth selection of Lazy Learning is adaptive. The minimum number of neighbors was 6, because at last $(n+1)(n+2)/2$ samples are needed to compute a quadratic polynomial function. In our case, n was equal to $\dim(\mathbf{a}) = 2$. The maximum number of neighbors was varied between 6 and 15 to verify the sensitivity of the EVPI estimate. The results of this sensitivity analysis over EVPI are shown in Figure 6.3.1. Observe that EVPI is insensitive to the maximum number of neighbors when this parameter is bigger then eight. However, note that there are still many ways to setup the Lazy Learning smoother and thus, caution is required when interpreting results.

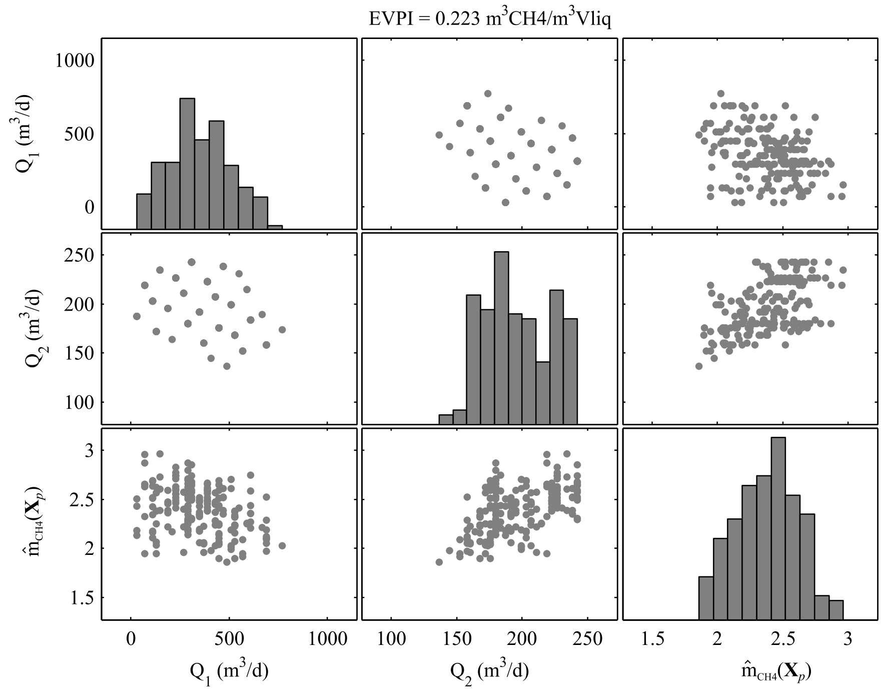


Figure 6.3.2: Estimated distribution of $\hat{m}_{CH4}(\mathbf{X}_p)$, and distributions of the relative optimal actions Q_1 and Q_2. The maximum number of neighbors for the Lazy Learning smoother is set to twelve.

Figure 6.3.2 represents the distributions of $\hat{m}_{CH4}(\mathbf{X}_p)$ and the relative optimal actions $\mathbf{a} = [Q_1,\, Q_2]$ when the maximum number of neighbors is twelve. As expected, significant positive correlation (+0.53) is observed between the beet energy crop substrate inflow rate, Q_2, and the maximum methane production conditional on perfect information, $\hat{m}_{CH4}(\mathbf{X}_p)$. This confirms the strong influence of the co-substrate over the methane production. Contrary, a slight negative correlation (-0.31) is present between the pig sludge inflow, Q_1, and $\hat{m}_{CH4}(\mathbf{X}_p)$. On average, the range of optimal actions within perfect information for Q_1 was [0 750] m^3/d, while for Q_2 was [125 250] m^3/d. The estimated EV|PI value was 2.38 m^3CH4/m^3Vliq/d, while the estimated EV was 2.16 m^3CH4/m^3Vliq/d. Note that this EV (i. e. max$\boldsymbol{\mu}^U_{CH4}$) is not equal to the GPR-approximated expected methane yield (i. e. max$\boldsymbol{\mu}^U_{CH4,GP}$) of 2.30 m^{3}CH$_4$/m^{3}Vliq/d because only discrete actions were considered. The estimate for EVPI was 0.22 m^3CH4/m^3Vliq/d, which can be more informative to a policy maker if converted to a monetary value. For example, assume the renewable-energy bonus is 0.135 €/kWh_{el} (Spanish scenario 2011), the internal consumption of electricity for a biogas plant is roughly 10% of the total electricity produced, the turbine efficiency of electricity conversion is 35% (kWh_{el}/kWh_{th}), and 1 m^3 of methane produces 10 kWh_{th} of thermal energy. After one year of operation, the expected monetary value gain for the SAVA biogas plant (6,000 m^3) would be 207,279 € if perfect information were available. However, to complete our VPI analysis, we should discount the cost of money (i. e. time value of money) from the estimated EMVPI. Obviously, the motivation for the development of the new laboratory technique increases if other similar biogas plants exist or are going to be built before the new lab-procedure project ends.

There are two important drawbacks to consider in the above analysis. The first is computational: bias and confidence intervals estimates for EVPI are not calculated as it is required (Oakley et al., 2010). To grantee an unbiased and precise estimation of $\hat{m}_{CH4}(\mathbf{X}_p)$, apart of increasing the number of inner level samples, we could increase the number of action samples. Because Lazy Learning returns an estimation of the goodness-of-fit for each local model, an adaptative sampling procedure could be performed to speed-up convergence. Due to the small sample size (limiting computation time) the above partial EVPI estimate is more an indicator rather than a reliable absolute result. Thus, further work is necessary in order to improve its credibility, since the role of smoothing techniques could have an instrumental role in estimating EVPI and its use requires further research. The second drawback is the assumption that the acquired information is perfect. In reality, perfection is difficult to achieve. However, the framework can be adopted to account for imperfect (or sample) information too.

Bibliography

Bontempi, G., Birattari, M., Bersini, H., 1997. Lazy learning for local modelling and control design. International Journal of Control 72, 643–658.

Hammersley, J. M., 1960. Monte carlo methods for solving multivariable problems. Annals of the New York Academy of Sciences 86 (3), 844–874.

Oakley, J. E., Brennan, A., Tappenden, P., Chilcott, J., 2010. Simulation sample sizes for Monte Carlo partial EVPI calculations. Journal of Health Economics 29 (3), 468–477.

Tappenden, P., Chilcott, J. B., Eggington, S., Oakley, J.and McCabe, C., 2004. Methods for expected value of information analysis in complex health economic models: developments on the health economics of interferon-β and glatiramer acetate for multiple sclerosis. Tech. Rep. 27, Health Technology Assessment.

Appendix B. VP-code and -interface

```
%% Model info
clear
% Define and describe the model
Code.modelname='AMH1';
Code.infoauthor={'Zivko Juznic-Zonta';['Copyrigth,' date];...
                 'Mollet del Valles, BCN, Spain';'UPC/GIRO-CT'};
Code.infomodel={'Simple modified Haldane model (Andrews, 1968) for AD'};
% Load the proyect from xls
S=loadxls(Code);
% Transform in symbolic S
S=sysinfo(S);
S=mcmcparam(S);
% Path network diagram (if stoich is variable, pathnet cannot be computed)
Adj_s=pathnet(S,Code);
% Save the data of the model
eval(['save ' Code.modelname '.mat']);

%% Write/Compile the ODE system

% Build the Code structure
Code=buildCode(S,Code);
% Write the C S-function for simulink
CmexModel(Code,true);
%MfileModel(Code,'sundials',S.xls.Mass);
% Generete the data class of the model
%clear PNM
PNM=modelclass(S,Code);
%clear Code data

%% Global SA with GP

% Plat the scatter plot matrix
NXlhs = 16;
%str={'\mu' 'K_s' 'K_i' 'X_{bio}(0)'};
str={'\mu' 'K_s' 'X_{bio}(0)'};
[Xlhs,SSlhs] = SAsslhs(PNM,NXlhs^2,str);

%% Optimization with GOm

% Use the negative log likelihood (MAP) or the LS-estimate.
likemisfit = true;
% Minimize the misfit function
ResultsGO = optimizationGO(PNM,likemisfit);

%% Data vs Model Prediction

outnames={'yXbio' 'yQm' 'yCODtot'};
PNMopt=PNMplot(PNM,[],outnames,'r');

%% Contour Plot for the Objective Function
```

```matlab
% Set the sensitivity range in %
settings.Range = 30;
% Sensitivity grid resolution in %
settings.step = 2;
% Upper and lower bounds option
settings.posparam = true;
% Calculate the sensitivity contour
outStr = fcnSensitivityRun('ssobjectiveGO',ResultsGO.xbest,...
    settings,PNM,likemisfit);
% Contour plots setting
outStr.setts.CplotRange = [90:2:100];
% Plot the countour
fcnMLPlot(outStr)

%% Sample from the posterior

% Run the DRAM-sampler (configure the sampler)
ResultsMCMC = optimizationMCMC(PNM);

% Burn-in and thin the MC-sample
burnin=5000; p=1;
chain   =ResultsMCMC.chain(burnin:p:end,:);
s2chain =ResultsMCMC.s2chain(burnin:p:end,:);
sschain =ResultsMCMC.sschain(burnin:p:end,:);

% Check autocorrelation of the chains
for i=1:size(chain,2)
    h(i)=plot(acf(chain(:,i),50));hold on
end
hold off

% Visualy inspect the chains
figure(1), mcmcplot(chain,[],ResultsMCMC,'chainpanel',1)
figure(2), mcmcplot(chain,[],ResultsMCMC,'pairs',1)
figure(3), mcmcplot(chain,[],ResultsMCMC,'hist')

% Summarizing statistics
[s1,s2] = chainstats(chain,ResultsMCMC);

% Correlation matrix from the MC-sample
ResultsMCMC.corr=corrcov(ResultsMCMC.cov);
RHOp = corr(chain,'type','Pearson')
RHOs = corr(chain,'type','Spearman')

%% Linear confidence reagions

% Check the auto-correlation in residuals before computing the
% parameter-uncertainty
[sse,res] = ssobjectiveGO(ResultsGO.xbest,PNM,likemisfit);
```

```
 % Estimated mode value of the model-parameters
if likemisfit x_mode = ResultsGO.xbest;
else x_mode = ResultsGO.xbest(1:end-1); end

% Auto-correlation function plot
figure, max_lags = 10; % Number of lags
stem(0:max_lags,acf(res,max_lags),'Color','k'),hold on
plot([0 max_lags+.5], 1.96/sqrt(length(res))*ones(1,2),'--k' ); % upper CI
plot([0 max_lags+.5], -1.96/sqrt(length(res))*ones(1,2), '--k'); % lower CI
xlim([0 max_lags+1]),xlabel('Lags'),ylabel('ACF'),ylim([-1,1])

% Check the normality of the residuals
figure, normplot(res)
%%
% Function over which to estimate the Hessian matrix. Note that the
% precision of the ode solver should be high (see simulation.m)
J = @(x)ssobjectiveGO(x,PNM,likemisfit);
% Estimate the Hessian with the "Adaptive Robust Numerical Differentiation"
% toolbox provided by D'Errico (2006). Note that parameters in the
% derivest( ) should be set. We used DerivativeOrder = 1, MethodOrder = 4,
% Style = 'central', RombergTerms = 2 and MaxStep = 0.9
[H_est,err] = hessianest(J,x_mode);
% cond(X,p) near 1 indicate a well-conditioned matrix.
cond_num = cond(H_est);
% Covariance matrix (Marsili-Libelli2002). If the objective function is a
% sum-of-squares (SS) than the covariance matrix C=inv(H(SS(x_opt))), but
% if the misfit function is a log-likelihood than C=inv(-H(L(x_mode))).
% Note that FIM=2*H and an estime of the varance for the SS misfit is
% SS(x_mode)/(n-p).

n = length(res); % Number of measurements
p = numel(x_mode); % Number of parameters
v = n-p; % Degree of freedom

% There is no difference between the two methods below
if likemisfit % Negative log-likelihood
    Ch_est = 2*inv(H_est);
else % LS-estimate
    s2 = sse/v;
    Ch_est = 2*s2*inv(H_est);
end

% Confidence intervals at 95%
delta = sqrt(diag(Ch_est)) * tinv(1-0.05/2,v);
ci = [(x_mode(:) - delta) (x_mode(:) + delta)];

%% Bayesian and Linear ellipses

% Linear confidence region from the Hessian (red)
C(:,:,1) = ResultsMCMC.cov;
```

```
% Linear c.r. from the MC-sample (blue)
if likemisfit % s2 is not considered
    C(:,:,2) = Ch_est(1:end-1,1:end-1);
    Mu = [x_mode(1:end-1);x_mode(1:end-1)]; % Ellipses centers
else
    C(:,:,2) = Ch_est;
    Mu = [x_mode;x_mode];
end
% Plot the ellipses and the MCMC samples
ellipse_pairs(Mu,C,ResultsMCMC.names,{'r','g'},0.95,chain);
```

```
function S = loadxls(Code)
% Load the proyect from xls

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Get the address where xls is stored
pathname = [cd '\Models\' Code.modelname '.xls'];

% For each xls sheet get the relative data
[ans,ans,input]   = xlsread(pathname,'input');
[ans,ans,param]   = xlsread(pathname,'param');
[ans,ans,stoich]  = xlsread(pathname,'stoich');
[ans,ans,rates]   = xlsread(pathname,'rates');
[ans,ans,func]    = xlsread(pathname,'func');
[ans,ans,output]  = xlsread(pathname,'output');
[ans,ans,dataxls] = xlsread(pathname,'data');
[ans,ans,graph]   = xlsread(pathname,'graph');

% Get the stoichometric matrix as cell structure
stoich       = nansetempty(stoich);
S.xls.stoich = stoich(2:end,2:end-1);
S.xls.Mass   = cell2mat(stoich(2:end,end));

% Get the model parameters:
param(2:end,4:end) = string2double(param(2:end,4:end));
S.xls.param        = param;
S.xls.rates        = rates;
S.xls.func         = func;
S.xls.input        = input;
S.xls.output       = output;
S.xls.dataxls      = dataxls;
S.xls.graph        = graph;
end

function Mzeros = nansetempty(Mnan)
% Change Nan for zeros in the stoichometric matrix
Mzeros = Mnan;
for i = 1:size(Mnan,1)
    for j = 1:size(Mnan,2)
        v = Mnan(i,j);
        if isnan(v{:})
            Mzeros{i,j} = 0;
        end
    end
end
end
```

```
function [Xlhs,SSlhs]=SAsslhs(PNM,Nlhs,param_name)
% I/O for probabilisti sensitivity analysis

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Chose the function to evaluate.
% 'MCMC' is the sum of squares for each model output relative to data
% 'misfit' is the likelihood or the mean square of residuals
% 'output' is a given output from the model
ss_type = 'misfit';
% Output index
if strcmp(ss_type,'output')
    idx_out = 1; % Qch4 = 1, Eff_CODrem = 3
end

% Number of samples for the LHS
if nargin<2
    Nlhs=200;
end

% Location of the parameters to be optimized
optidx    =targetvector(PNM,'global');
opt_param =PNM.Parameters(optidx(1,:));
% Open the PNM data structure of the parameters (DRAM toolbox - Laine)
[names,value,parind,local,upp,low] = ...
    openparstruct(opt_param,length(PNM.Data));

% First principle models have positive parameters
low(low==0)   =1e-10;
upp(upp==inf) =1e10;

% LHS design for sensitivity analysis
Xlhs_0 =lhsdesign(Nlhs,length(low));
Xlhs   =Xlhs_0.*repmat(abs(upp-low),Nlhs,1)+repmat(low,Nlhs,1);

% Simulate the model in order to obtain the SS function
if ~strcmp(ss_type,'MCMC')
    SSlhs=zeros(Nlhs,1);
end

for i=1:Nlhs
    switch ss_type
        case 'MCMC'
            ss = ssobjectiveDRAM(Xlhs(i,:),PNM,local,parind);
            SSlhs(i,:) = ss;
        case 'misfit'
            % Mean sum of squares function (modify ssobjectiveGO!)
            %mse = ssobjectiveGO(Xlhs(i,:),PNM); SSlhs(i)=sqrt(mse);
            % Normal likelihood function
            std_ref = 2.8; % Should provide the std for measur. errors
            param = [Xlhs(i,:),std_ref];
            nloglike = ssobjectiveGO(param,PNM); SSlhs(i)=nloglike;
        case 'output'
            % Change the parameters to be optimized inside the Smodel↙
structure
            PNMnew = changeparamGO(Xlhs(i,:),PNM);
            % Get the model simulations for the i-th batch experiment
            ibatch = 1; SampleTime = [0 500]; % steady-state
            [t,x,y] = simulation(PNMnew,ibatch,SampleTime);
            SSlhs(i) = y(end,idx_out); % output variable at steady-state
            % Note. If we use FAST (Saltelli) rutine, then time points can
            % be considered.
    end
end

% Plot the param. against the SSlhs value
if ~strcmp(ss_type,'MCMC')
    if nargin==3
        nsub = ceil(sqrt(size(Xlhs,2)));
        [ans,idx_minSS] = sort(SSlhs);
        % Find the approx minima
        idx_minSS = idx_minSS(1:1);
        minSS = SSlhs(idx_minSS);
        for i=1:size(Xlhs,2)
            subplot(nsub,nsub,i)
            plot(Xlhs(:,i),SSlhs,'ok','MarkerFaceColor',...
                [.7 .7 .7],'MarkerSize',4)
            minX = Xlhs(idx_minSS,i);hold on
            plot(minX,minSS,'ok','MarkerFaceColor',...
                [0 0 0],'MarkerSize',6)
            xlabel(param_name{i}),ylabel('MSE')
        end
    end
end
% Get the name of the model
PNM.Code.modelname;
% Save in a txt file for the O'Hogan's SA-GP application
cd( [cd '/Models'])
eval(['save ' PNM.Code.modelname '_SSlhs.txt SSlhs -ascii -double -tabs'])
eval(['save ' PNM.Code.modelname '_Xlhs.txt  Xlhs  -ascii -double -tabs'])
% Return to the current directory
cd('..')
```

```
function [t,x,y,u] = simulation(PNM,nbatch,SampleTime,InitialConditions)
% Simulation of the model

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

if nargin < 2
    nbatch =1;
    SampleTime = 1;
elseif nargin <3
    SampleTime = 1;
end

% Open the param_mcmc data structure
[names,value,parind,local] = ...
    openparstruct(PNM.Parameters,length(PNM.Data));
% Get the parameters of the model relative to the batch experiment
theta=value(ismember(local,[0 nbatch]));

% Time vector over which the estimations should be performed
if SampleTime == 0 % Discrete
    tsim=PNM.Data{nbatch}.ydata(:,1);
elseif SampleTime == 1 % Adaptative
    tsim=[min(PNM.Data{nbatch}.ydata(:,1)),max(PNM.Data{nbatch}.ydata(:,1))];
else
    tsim=SampleTime;
end

% Inflow vector
if iscell(PNM.Inflow)
    simin=PNM.Inflow{nbatch};
else
    simin=PNM.Inflow;
end

% Parameters
Parameters=theta(1:(end-PNM.Code.N_states));

% Initial conditions for the states
if nargin < 4
    InitialConditions=theta(end-PNM.Code.N_states+1:end);
end
% Solver options
options = simset('solver','ode15s','Reltol',1e-8,'AbsTol',1e-8,...
                 'SrcWorkspace','current','InitialStep',1e-10);
% Model name
model=[PNM.Code.modelname 'sim'];
```

```
% Simualate the model. Reduce the precision of the ode solver if
% singularity in solution accrues.
try
    [t,x]=sim(model,tsim,options);
    u=siminflow;
    y=simoutflow;
catch ME1
    try
    options = simset('solver','ode15s','Reltol',1e-6,'AbsTol',1e-6,...
                 'SrcWorkspace','current','InitialStep',1e-6,'Refine',1);
    [t,x]=sim(model,tsim,options);
    u=siminflow;
    y=simoutflow;
    catch ME2
        options = simset('solver','ode15s','Reltol',1e-3,'AbsTol',1e-6,...
                 'SrcWorkspace','current','InitialStep',1e-3,'Refine',1);
        [t,x]=sim(model,tsim,options);
        u=siminflow;
        y=simoutflow;

    end
end
```

```
function PNMnew=changeparamDRAM(value,PNM,local,parind)
%Change the parameters for DRAM sampler

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Initialize
PNMnew=PNM;
local_target=local(parind)';
value_target=value(parind);

% Location of the parameters to be optimized
optidx    =targetvector(PNM,'all');
opt_param =PNM.Parameters(optidx(1,:));
% Change the target parameters that are not local
x_opt = value_target(local_target==0);
for i = 1:length(x_opt)
    opt_param{i}{2}=x_opt(i);
end
PNMnew.Parameters(optidx(1,:))=opt_param;

% Change the parameters that are local
if ~isempty(find(optidx(2,:)==1,1))
    opt_param=PNM.Parameters(optidx(2,:));
    x_opt=value_target(local_target~=0);
    p=sum(optidx(2,:),2);
    if p>1
        A=mat2cell(reshape(x_opt,length(opt_param),p)',ones(1,p),p);
    else A={x_opt};
    end
    for i=1:length(opt_param)
        opt_param{i}{2}=A{i};
    end
    PNMnew.Parameters(optidx(2,:))=opt_param;
end
```

```matlab
function PNMnew=changeparamGO(x_opt,PNM)
%Change the parameters for the GO estimation

PNMnew=PNM; % Initialize PNMnew

% Change the parameters to be estimamted by GO
c=0; % counter for x_opt
for i=1:length(PNM.Parameters)
     % Check if the parameter is a target, but not local
    if PNM.Parameters{i}{1,7}==true && PNM.Parameters{i}{1,8}==false
        c=c+1;
        PNMnew.Parameters{i}{1,2}=x_opt(c);
        PNMnew.Parameters{i}{1,5}=x_opt(c);
    end
    % Check if the parameter is a target and is local
    if PNM.Parameters{i}{1,7}==true && PNM.Parameters{i}{1,8}==true
        x_local=zeros(1,length(PNM.Data));
        % Put the local values for parameter i-th in x_local
        for j=1:length(PNM.Data)
            c=c+1;
            x_local(j)=x_opt(c);
        end
            % Change the local x_opt with the new values
            PNMnew.Parameters{i}{1,2}=x_local;
            PNMnew.Parameters{i}{1,5}=x_local;

    end

end

end
```

```
function optidx=targetvector(PNM,typeGO)
% Get a vector of booleans that indicate which parameters has to be
% optimazied by the global optimization (GO) algorithm

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

if nargin < 2
    typeGO = 'all';
end
% Initialize
N_param =size(PNM.Parameters,1);
optidx  =zeros(2,N_param);
for i=1:N_param
    % Only the target parameters could be estimated by the GO since the
    % local parameters should be known, or at last their mean value should
    % be known
    switch typeGO
        case 'global'
            if PNM.Parameters{i}{1,7}==true && ~PNM.Parameters{i}{1,8}==true
                optidx(1,i)=true;
            else
                optidx(1,i)=false;
            end
        case 'all'
            % All the target parameters could be estimated
            if PNM.Parameters{i}{1,7}==true
                optidx(1,i)=true;
            else
                optidx(1,i)=false;
            end
        otherwise
            error('Should assign if all or only global parameters has to be↙
estimated.')
    end
end
% The second row indicates the presence of a parameter that has to be
% estimated by the MCMC, but is local. The GO estimation will assume
% that the initial value of this parameter is fixed on the mean value of
% N(mu,sig) relative to MCMC rutine
if PNM.Parameters{i}{1,7}==true && PNM.Parameters{i}{1,8}==true
    optidx(2,i)=true;
else
    optidx(2,i)=false;
end
optidx=logical(optidx);
end
```

```
function ss = ssobjectiveDRAM(value,PNM,local,parind)
% Sum of squares for DRAM sampler

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Change the parameters
PNMnew = changeparamDRAM(value,PNM,local,parind);

% Get the number of experiments
nbatch = length(PNM.Data);

% Initialize the SSE vector
ss = 0;
% Get the indexes of the measured model outputs
[ans,idx_out,idx_meas]=intersect(PNM.Output.Output_names,PNM.Measures);
% Sort the outputs as in the xls file (data sheet)
[ii,II]=sort(idx_meas);

% Set the norm type. L1 is more robust over outliers but could have
% multiple modes
p=2;  %L2
%p=1; %L1

for ibatch = 1:nbatch

    % Do not perform simulation over experiments with weights = 100
    if ~all(all(PNM.Data{ibatch}.weights(:,2:end)==100))
        % Get the data for the i-th batch experiment
        datai = PNM.Data{ibatch};
        % Get the measured data and remove 1st "time" columns
        ydata  = datai.ydata(:,2:end,:);
        % Get the model simulations for the i-th batch experiment
        [ans,ans,y] = simulation(PNMnew,ibatch,0);
        ymodel=y(:,idx_out);
        ymodel=ymodel(:,II);
        % Maximum number of repetitions
        Max_rep=size(ydata,3);

        % Squeeze the multidim. arrays of measurements into a 2Dmatrix
        ydata=catmultidim(1,ydata);
        ymodel=repmat(ymodel,Max_rep,1);

        % Scale parameter
        %S=PNM.Scale{ibatch}; % for every batch
        S=repmat(PNM.Scale(:)',size(ydata,1),1);
        S=S(:,idx_meas);
        if isfield(datai,'weights')
            % Get the weights
            weights = 1./datai.weights(:,2:end,:);
            weights = catmultidim(1,weights);
            % Multiply the weights with the relative measurements and
            % compute the residuals
            res = weights.*(ydata-ymodel)./S;
            % Weighted sum of squares
            ss = ss + nansum(abs(res).^p); % Normal pdf likelihood
            %nu = 3;
            %ss = ss + (nu+1)*nansum(log(nu+abs(res).^2)); % t-student
        else
            % Scaled residuals
            res=(ydata-ymodel)./S;
            % Sum of squares
            ss = ss + nansum(abs(res).^p);
            %nu = 3;
            %ss = ss + (nu+1)*nansum(log(nu+abs(res).^2)); % t-student
        end
    end
end

% The output Y variances could be estimated (MAP) or fixed known in order
% to account for the different scales of magnitude. In Evans 2001 it is
% proposed  to use as a scale factor for the misfit funtion the maximum
% of a time series of measured data, or if the data has outliers, the 90%
% percentile. More over, the generalized gaussian distribution is proposed,
% as a misfit function: 1/b*abs((Ysim^b-Ymeas^b)/S^b)^p. If p=2 and b=1
% than a gaussian distribution arise. A usual choise is p=2 and b=1/2. We
% should take care that the model parameter estimation is not too sensible
% to the misfit function.
```

```
function Results = optimizationGO(PNM,likemisfit)
% Scatter Search optimization and relative options

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Objective funtion to minimize
problem.f='ssobjectiveGO';

% Location of the parameters to be optimized
optidx=targetvector(PNM,'all'); %Only global param can be optimized by GO
opt_param=PNM.Parameters(optidx(1,:));

% Open the PNM data structure of the parameters
[names,value,parind,local,upp,low] = ...
    openparstruct(opt_param,length(PNM.Data));

% First order model parameters should be positive and non-inf
low(low==0)  =1e-10;
upp(upp==inf)=1e10;

if likemisfit
    % Negative log-Likelihood minimization (MAP-estimate)
    sigma_0 = 1; sigma_L = 1e-1; sigma_U = 10; % std measur. errors
    problem.x_0 = [value' sigma_0];
    problem.x_L = [low    sigma_L];
    problem.x_U = [upp    sigma_U];
else
    % Mean squared error minimization (LS-estimate)
    problem.x_0 = value';
    problem.x_L = low;
    problem.x_U = upp;
end

% SS-optimization options
opts.local.finish='fminsearch';
opts.local.solver=0; % 'fmincon', 'fminsearch', 'solnp', 'n2fb'
opts.maxtime=1e4;
opts.maxeval=3e3; % Number of max evaluations of ObjFun (5e3*1.8/3600=2.5h)
%opts.log_var = 1:length(value)+1;
%opts.plot = 1;    % plot convergence curves in real time
opts.ndiverse = 500;

% Run the global optimization. ess_kernel() is the scatter-search
% optimization rutine of Rodriguez-Fernandez (2006) from the GOm toolbox.
Results = ess_kernel(problem,opts,PNM,likemisfit);

% Save in Results structure some usefull informations
```

```
Results.problem = problem;
Results.opts    = opts;

% Print the estimates
for i=1:length(opt_param)
    disp([opt_param{i}{1} ' = ' num2str(Results.xbest(i))])
end
```

```
function Results = optimizationGO(PNM,likemisfit)
% Scatter Search optimization and relative options

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Objective funtion to minimize
problem.f='ssobjectiveGO';

% Location of the parameters to be optimized
optidx=targetvector(PNM,'all'); %Only global param can be optimized by GO
opt_param=PNM.Parameters(optidx(1,:));

% Open the PNM data structure of the parameters
[names,value,parind,local,upp,low] = ...
    openparstruct(opt_param,length(PNM.Data));

% First order model parameters should be positive and non-inf
low(low==0)  =1e-10;
upp(upp==inf)=1e10;

if likemisfit
    % Negative log-Likelihood minimization (MAP-estimate)
    sigma_0 = 1; sigma_L = 1e-1; sigma_U = 10; % std measur. errors
    problem.x_0 = [value' sigma_0];
    problem.x_L = [low    sigma_L];
    problem.x_U = [upp    sigma_U];
else
    % Mean squared error minimization (LS-estimate)
    problem.x_0 = value';
    problem.x_L = low;
    problem.x_U = upp;
end

% SS-optimization options
opts.local.finish='fminsearch';
opts.local.solver=0; % 'fmincon', 'fminsearch', 'solnp', 'n2fb'
opts.maxtime=1e4;
opts.maxeval=3e3; % Number of max evaluations of ObjFun (5e3*1.8/3600=2.5h)
%opts.log_var = 1:length(value)+1;
%opts.plot = 1;    % plot convergence curves in real time
opts.ndiverse = 500;

% Run the global optimization
Results = ess_kernel(problem,opts,PNM,likemisfit);

% Save in Results structure some usefull informations
Results.problem = problem;
Results.opts    = opts;

% Print the estimates
for i=1:length(opt_param)
    disp([opt_param{i}{1} ' = ' num2str(Results.xbest(i))])
end

% When using n2fb (or dn2fb) and lsqnonlin as local solvers, the objective
% function value must be formulated as the square of the sum of differences
% between the experimental and predicted data.
```

```
function PNM=PNMplot(PNMold,Results,outnames,type,batchexp)
% Plot the outputs of the model

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Do you want to plot?
fig=true;

if ~isempty(Results)
    PNM = changeparamGO(Results.xbest,PNMold);
else
    PNM=PNMold;
end

if nargin<4
    type='b';
end
if nargin<5
    batchexp=1:length(PNM.Data);
end

% Sampling time for simulation
SampleTime=[0,470];
%SampleTime=0;

% Find the measured states
idx_meas=multistrcmp(PNM.Output.Output_names,PNM.Measures);
idx_plot=multistrcmp(PNM.Output.Output_names,outnames);
[idx,idxl]=multistrcmp(PNM.Output.Output_names(idx_plot),PNM.Measures);

iii=0;
for ibatch = batchexp;

    % Get the data for the i-th batch experiment
    datai = PNM.Data{ibatch};
    ydata  = datai.ydata(:,1:end,:);
    ydata=catmultidim(1,ydata);

    % Get the model simulations for the i-th batch experiment
    [t,x,ymodel] = simulation(PNM,ibatch,SampleTime);
    ymodel=ymodel(:,idx_plot);
    % Save the simulation outputs into the PNM structure
    PNM.Data{1,ibatch}.ydatasim=[t,ymodel];

    if fig
        iii=iii+1; figure(iii)
        % Plot the single variables of the batch experiment
        smax=length(idx_plot);
        for s=1:smax
            subplot(smax,1,s)
            if idxl(s)
                count = find(strcmp(outnames(s),PNM.Measures));
                hold on
                plot(ydata(:,1),ydata(:,1+count),...
                    'o','MarkerFaceColor',type,...
                    'MarkerEdgeColor',[.5,.5,.5],...
                    'MarkerSize',2)
            end
            hold on
            plot(t,ymodel(:,s),type)
            %ylabel( {PNM.Output.Output_names{idx_plot(s)} ;['(' PNM.Output.↙
Output_units{idx_plot(s)} ')']})
            ylabel( {PNM.Output.Output_names{idx_plot(s)}} )
            if s==1
                %title(['Experiment No.' num2str(ibatch)])
            end
            a=min(ymodel(:,s));
            if a<0, a=0; end
            b=ceil(max(ymodel(:,s))+.1*max(ymodel(:,s)));
            if b<=0 || isnan(b), b=0.1; end
            ylim([a b])
            %ylim([0 10])
            %xlim([SampleTime(1) SampleTime(end)])
        end
        xlabel('Time')
    end
end
end
```

```
function ResultsMCMC=optimizationMCMC(PNM,MCMCresults0)
% DRAM sampling from the parameter posterior

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% If available, set the std measur. error
s2=2.5^2;

% Bayesian Analysis for all models

PNM.ModelMCMC.ssfun   = @ssobjectiveDRAM;
%PNM.ModelMCMC.ssfun   = @ssobjectiveDRAM_gpr; % Not ready yet

PNM.ModelMCMC.sigma2  = s2; % initial error variance
%PNM.ModelMCMC.S20 = [];    % initial error variance (multiple measur.)
%PNM.ModelMCMC.N0  = [];    % prior (invchisq) weight for sigma2

PNM.OptionsMCMC.method      = 'dram';% adaptation method (mh,am,dr,dram)
PNM.OptionsMCMC.nsimu       = 60000; % n:o of simulations
PNM.OptionsMCMC.burnintime  = 5000;  % Burn-in
%PNM.OptionsMCMC.qcov        = [];    % proposal covariance
PNM.OptionsMCMC.adaptint    = 500;   % adaptation interval
PNM.OptionsMCMC.printint    = 200;   % how often to show info accept.ratios
PNM.OptionsMCMC.verbosity   = 1;     % show output in Matlab window
PNM.OptionsMCMC.waitbar     = 1;     % show garphical waitbar
PNM.OptionsMCMC.updatesigma = 1;     % update error variance
PNM.OptionsMCMC.stats       = 1;     % save extra statistics in results

% Run the MCMC chain
if nargin<2
    [ResultsMCMC,chain,s2chain,sschain]=mcmcrun(PNM.ModelMCMC,PNM,...
        PNM.Parameters,PNM.OptionsMCMC);
else
    [ResultsMCMC,chain,s2chain,sschain]=mcmcrun(PNM.ModelMCMC,PNM,...
        PNM.Parameters,PNM.OptionsMCMC,MCMCresults0);
end

% Store the main results from MCMC
ResultsMCMC.chain=chain;
ResultsMCMC.s2chain=s2chain;
ResultsMCMC.sschain=sschain;
```

```
function ks_pair = ellipse_pairs(mu,C,names,style,conf,chain)
% Confidence ellipses relative to paired parameters

% If MC-sample is provided then a comparison is computed between the
% posterior distribution and its relative linear approximation given by C.
% Two-sample Two-diensional Kolmogorov-Smirnov Test is used to assess the
% goodness of the linear approx to the posterior.

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

skip=1; % Thin the MC-sample points
[n,p,z] = size(C);
count = 0;

if p>10
    error('too many pairs')
end

% "Subplot" axes with adjustable gaps and margins
h = tight_subplot(p-1,p-1,[.03 .02],[.1 .1],[.1 .01]);

%clf
for j=2:p
    for i=1:j-1
        if p==2
            h=gca;
        else
            axes(h((j-2)*(p-1)+i));
            %h=subplot(p-1,p-1,(j-2)*(p-1)+i);
        end

        for k=1:z
            Csub=[C(i,i,k),C(i,j,k);C(j,i,k),C(j,j,k)];
            error_ellipse(Csub,'mu',[mu(k,i),mu(k,j)],...
                'style',style{k},'conf',conf);hold on
        end

        if nargin == 6
            panellims(chain(:,i),chain(:,j),2,[],0); hold on
            c=[.3,.3,.3]*1;
            plot(chain(1:skip*2:end,i),chain(1:skip*2:end,j),'.',...
                'MarkerFaceColor',c,'MarkerEdgeColor',c,...
                'MarkerSize',2,'Color',[1 1 1]*.5); hold on;
            plot(mu(k,i),mu(k,j),'+k','MarkerSize',6);
            xlim([(min(chain(:,i))),(max(chain(:,i)))])
            ylim([(min(chain(:,j))),(max(chain(:,j)))])
        end
        if nargout==1
            count=count+1;
            % Posterior sample
            r_1 = chain(1:skip:end,[i j]);
            % Multi-variable normal distribution approximation
            r_2 = mvnrnd([mu(1,i),mu(1,j)],...
                [C(i,i,1),C(i,j,1);C(j,i,1),C(j,j,1)],length(r_1));
            %Two-sample Two-diensional Kolmogorov-Smirnov Test
            [H, pValue, KSstatistic] = kstest_2s_2d(r_1, r_2);
            ks_pair(count,:)=[i,j,KSstatistic, pValue, H];
            text(0.05,0.9,['K-S=' num2str(KSstatistic,2)],...
                'Units','normalized ')
        end

        drawnow
        if j~=p
            set(h((j-2)*(p-1)+i),'xtick',[])
        end
        if i~=1
            set(h((j-2)*(p-1)+i),'ytick',[])
        end
        if i==1 & nargin>2 & ~isempty(names)
            ylabel(names{j})
        end
        if i==j-1 & nargin>2 & ~isempty(names)
            if p==2
                xlabel(names{i});
            else
                title(names{i})
            end
        end
    end
end
```

```
function S=sysinfo(S)
% Transform in symbolic S

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Get the names of the states,rates,parameters,...
S.States_name    =S.xls.graph(2:end,1);
S.Color_name     =S.xls.graph(2:end,2);
S.Rates_name     =S.xls.rates(:,1);
S.Parameters_name=S.xls.param(2:end,3);
S.Output_name    =S.xls.output(:,1);
S.Functions_name =S.xls.func(:,1);
S.Input_name     =S.xls.input(2,2:end)';
S.Data_name      =S.xls.dataxls(1,3:((size(S.xls.dataxls,2)-2)/2+2))';

% Text structures for equations and units
[ans,rates_equations]  =parsedata(S.xls.rates(:,2));
[ans,func_equations]   =parsedata(S.xls.func(:,2));
[ans,output_equations] =parsedata(S.xls.output(:,2));
[ans,input_units]      =parsedata(S.xls.input(1,2:end)');
[ans,output_units]     =parsedata(S.xls.output(:,3));
[ans,parameters_units] =parsedata(S.xls.param(2:end,2));

% Only for adimensional units with no suplement information
parameters_units(strcmp(parameters_units,''))={'1'};

% Symbolic stoich matrix
S.symStoich=cell2sym(S.xls.stoich);
% Symbolic states
S.symStates=cell2sym(S.States_name);
% Symbolic rates
S.symRates=cell2sym(S.Rates_name);
% Symbolic parameters
S.symParam=cell2sym(S.Parameters_name);
% Symbolic funcions
S.symFunc=cell2sym(S.Functions_name);
% Symbolic input
S.symInputs=cell2sym(S.Input_name);
% Symbolic input
S.symOutputs=cell2sym(S.Output_name);

% Reactions equations. Those are used in the ode system for computations,
% with the corresponding equations for the rates and the functions. In this
% way we could resolve the sparcity of some dynamic systems, since the zero
% stoichometric elements are not considered in the computations.
S.ODE = S.symStoich*S.symRates;
```

```
% Express the reactions in text format
n=length(S.ODE); ODE_equations=cell(n,1);

for i=1:n
    ODE_equations{i}=char(S.ODE(i));
end
S.rates_equations  =rates_equations;
S.rates_direction  =S.xls.rates(:,end-1);
S.states_graph     =S.xls.graph(2:end,:);
S.func_equations   =func_equations;
S.output_equations =output_equations;
S.ODE_equations    =ODE_equations;
S.input_units      =input_units;
S.output_units     =output_units;
S.parameters_units =parameters_units;

% Simple chack over the states name
if any(~strcmp(S.states_graph(:,1),S.States_name));
    error('Names of the states in stoich and graph should be the same.')
end

% Compute the stoichometric coefficients functions
symarray = [S.Parameters_name;S.States_name;S.Rates_name;...
    S.Functions_name;S.Input_name];

for i=1:size(symarray)
    eval([ 'syms ' symarray{i} ]);
end

for i=1:size(S.symFunc) % Define symbolic equations for the rates
    if isreal(S.xls.func{i,2})
        eval([ S.xls.func{i,1} '=' num2str(S.xls.func{i,2}) ';']);
    else
        eval([ S.xls.func{i,1} '=' S.xls.func{i,2} ';']);
    end
end

% Compute the extended ode system of equations
S.fullODE = subs(S.ODE,S.Rates_name,rates_equations);

% Compute sensitivity trajectory functions and Jacobian
if true

    symarray = [S.Parameters_name;S.States_name;...
        S.Rates_name;S.Functions_name;S.Input_name];

    for i=1:size(symarray)
        eval([ 'syms ' symarray{i} ])
    end
    for i=1:size(S.symRates) % Define symbolic equations for the rates
        if isreal(S.xls.rates{i,2})
            eval([ S.xls.rates{i,1} '=' num2str(S.xls.rates{i,2}) ';']);
        else
            eval([ S.xls.rates{i,1} '=' S.xls.rates{i,2} ';']);
        end
    end

    for i=1:size(S.symFunc) % Define symbolic equations for the functions
        if isreal(S.xls.func{i,2})
            eval([ S.xls.func{i,1} '=' num2str(S.xls.func{i,2}) ';']);
        else
            eval([ S.xls.func{i,1} '=' S.xls.func{i,2} ';']);
        end
    end

    % Compute two times the substitution of sym expressions for functions
    % because there could be functions defined by using already defined
    % functions (e.g., pow)
    S.ODE_long=subs(S.fullODE,S.Functions_name,S.func_equations);
    S.ODE_long=subs(S.ODE_long,S.Functions_name,S.func_equations);

    % Sensitivity functions
    theta     =[S.symParam]; % parameters
    Num_ODE   =size(S.ODE_long,1);
    Num_theta =size(theta,1);
    Num_React =size(S.symStates,1);
    syms a
    SensFunc(Num_theta,Num_ODE)     =a;
    SensFunc_idx(Num_theta,Num_ODE) ={''};

    for i=1:Num_theta
        for j=1:Num_React
            SensFunc(i,j)=diff(S.ODE_long(j),theta(i),1);
            SensFunc_idx{i,j}=['d' char(S.symStates(j))...
                '/d' char(theta(i)) ];
        end
    end

    S.SensFunc     =SensFunc; % sensitivity function matrix
    S.SensFunc_idx =SensFunc_idx; % dstate/dtheta matrix

    % Jacobian matrix
    try % D[1]pow(x,a) should be evaluated
        S.Jacobian =jacobian(eval(S.ODE_long),S.symStates);
    catch  % no D[1]pow(x,a) is present
        S.Jacobian =jacobian(S.ODE_long,S.symStates);
    end
end
end
```

```
function P=cell2sym(X)
% Cell to symbolic representation

[r,c]=size(X);

syms P
P(r,c)=0;
for i=1:r
    for j=1:c
        if ~ischar(X{i,j})
            P(i,j)=X{i,j};
        else
            P(i,j)=sym(X{i,j});
        end
    end
end
end

function [numericArray,textArray] = parsedata(data)
% Parse data from raw cell array into a numeric array and a text
% cell array

% Input:
%        data: cell array containing data from spreadsheet
% Return:
%        numericArray: double array containing numbers from spreadsheet
%        textArray: cell string array containing text from spreadsheet


% ensure data is in cell array
if ischar(data)
    data = cellstr(data);
elseif isnumeric(data) || islogical(data)
    data = num2cell(data);
end

% Check if raw data is empty
if isempty(data)
    % Abort when all data cells are empty.
    textArray = {};
    numericArray = [];
    return
else
    % Trim empty leading and trailing rows
    % find empty cells
    emptycells = cellfun('isempty',data);
    nrows = size(emptycells,1);
    firstrow = 1;
    % find last of leading empty rows
    while (firstrow<=nrows && all(emptycells(firstrow,:)))
        firstrow = firstrow+1;
    end
    % remove leading empty rows
    data = data(firstrow:end,:);

    % find start of trailing empty rows
    nrows = size(emptycells,1);
    lastrow = nrows;
    while (lastrow>0 && all(emptycells(lastrow,:)))
        lastrow = lastrow-1;
    end
    % remove trailing empty rows
    data = data(1:lastrow,:);

    % find start of trailing NaN rows
    warning('off', 'MATLAB:nonIntegerTruncatedInConversionToChar');
    while (lastrow>0 && ~(any(cellfun('islogical', data(lastrow,:))))&& ...
            all(isnan([data{lastrow,:}])))
        lastrow = lastrow-1;
    end
    warning('on', 'MATLAB:nonIntegerTruncatedInConversionToChar');
    % remove trailing NaN rows
    data=data(1:lastrow,:);

    [n,m] = size(data);
    textArray = cell(size(data));
    textArray(:) = {''};
end

vIsNaN = false(n,m);

% find non-numeric entries in data cell array
vIsText = cellfun('isclass',data,'char');
vIsNaN = cellfun('isempty',data)|strcmpi(data,'nan')...
    |cellfun('isclass',data,'char');

% place text cells in text array
if any(vIsText(:))
    textArray(vIsText) = data(vIsText);
else
    textArray = {};
end
% Excel returns COM errors when it has a #N/A field.
textArray = strrep(textArray,'ActiveX VT_ERROR: ','#N/A');

% place NaN in empty numeric cells
if any(vIsNaN(:))
    data(vIsNaN)={NaN};
```

```
end

% extract numeric data
data = reshape(data,n,m);
rows = size(data,1);
m = cell(rows,1);
% Concatenate each row first
for n=1:rows
    m{n} = cat(2,data{n,:});
end
% Now concatenate the single column of cells into a matrix
numericArray = cat(1,m{:});


% trim all-NaN leading rows and columns from numeric array
% trim all-empty trailing rows and columns from text arrays
[numericArray,textArray]=trim_arrays(numericArray,textArray);

% ensure numericArray is 0x0 empty.
if isempty(numericArray)
    numericArray = [];
end
end

function [numericArray,textArray] = trim_arrays(numericArray,textArray)
% trim leading rows or cols
% if the string result has dimensions corresponding to a column or row of
% zeros in the matrix result, trim the zeros.
if ~isempty(numericArray) && ~isempty(textArray)
    [mn, nn] = size(numericArray);
    [ms, ns] = size(textArray);

    if ms == mn
        % trim leading column(textArray) from numeric data
        firstcolm = 1;
        while (firstcolm<=nn && all(isnan(numericArray(:,firstcolm))))
            firstcolm = firstcolm+1;
        end
        numericArray=numericArray(:,firstcolm:end);
    end

    if ns == nn
        % trim leading NaN row(s) from numeric data
        firstrow = 1;
        while (firstrow<=mn && all(isnan(numericArray(firstrow,:))))
            firstrow = firstrow+1;
        end
        numericArray=numericArray(firstrow:end,:);

        % trim leading empty rows(s) from text data
        firstrow = 1;
        while (firstrow<=ms &&...
                all(cellfun('isempty',textArray(firstrow,:))))
            firstrow = firstrow+1;
        end
        textArray=textArray(firstrow:end,:);
    end

    % trim all-empty-string trailing rows from text array
    lastrow = size(textArray,1);
    while (lastrow>0 && all(cellfun('isempty',textArray(lastrow,:))))
        lastrow = lastrow-1;
    end
    textArray=textArray(1:lastrow,:);

    % trim all-empty-string trailing columns from text array
    lastcolm = size(textArray,2);
    while (lastcolm>0 && all(cellfun('isempty',textArray(:,lastcolm))))
        lastcolm = lastcolm-1;
    end
    textArray=textArray(:,1:lastcolm);

    % trim all-NaN trailing rows from numeric array
    lastrow = size(numericArray,1);
    while (lastrow>0 && all(isnan(numericArray(lastrow,:))))
        lastrow=lastrow-1;
    end
    numericArray=numericArray(1:lastrow,:);

    % trim all-NaN trailing columns from numeric array
    lastcolm = size(numericArray,2);
    while (lastcolm>0 && all(isnan(numericArray(:,lastcolm))))
        lastcolm=lastcolm-1;
    end
    numericArray=numericArray(:,1:lastcolm);
end
end
```

```
function S=mcmcparam(Sold)
% Transform the parameters data to the DRAM toolbox format

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

S=Sold;
N_param=size(Sold.xls.param,1)-1; param_mcmc{N_param,1}=0;
for i=1:N_param
    param_mcmc{i}=Sold.xls.param(i+1,3:end);
end
S.param_mcmc=param_mcmc;
```

```matlab
function [Adj_s,stoich_matrix_val]=pathnet(S,Code,experiment)
% Path network diagram

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

if nargin<4
    % Batch experiment number for which the diagram is build
    experiment=1;
end

% Get the number of batch experiments from the data structure (incoplete)
%nbatch=size(data,2);
nbatch=1;
% Open the param_mcmc data structure
[names,value,parind,local] = openparstruct(S.param_mcmc,nbatch);
% Value of the parameters of the corresponding local batch experiment
value_batch=value(ismember(local, [0,experiment]));
% Get the index vector:[bio/chem param==1, stoich coeff==2, IC states==3]
idx=cell2mat(S.xls.param(2:end,1));
% Get only the stoich coeff
Stoich_val=num2cell(value_batch(idx==2))';
% Get the names of the stoich coeff
Stoich_name=S.Parameters_name(idx==2)';
% Compute the numeric stoich matrix
if ~isempty(Stoich_name)
    stoich_matrix_val=subs(S.symStoich,Stoich_name,Stoich_val);
else
    stoich_matrix_val=roundoff(double(S.symStoich),2);
end

% Adjont matrix (cell text structure). Comment which info should be
% considered in the graphviz plot
[Adj,Adj_net,arcdir]=stoich2adj(stoich_matrix_val,S);
%[Adj,Adj_net]=stoich2adj(stoich_matrix_val,S.Rates_name,S.stoich_matrix);

% Export a Graphviz .dot file of the model network
cd( [cd '/Models'])
graph2dot([Code.modelname '.dot'],Adj,Adj_net,arcdir,S);

% Adjacent matrix of the states
N = stoich_matrix_val; N(N~=0)=1;
Adj_s=triu(N*N');

% % Graph in matlab. No arc names are permited since the script has an error
% graph_to_dot(Adj_net, 'filename',[Code.modelname 'Matlab.dot'],...
%     'node_label',node_label,'node_colorfill',node_colorfill,...
%     'width',20,'height',20);
% %pathname='C:\Programmi\Graphviz2.26.3\bin\dot.exe';
% pathname='C:\Graphviz2.27\bin\dot.exe';
% figure,drawDot2matlab([Code.modelname 'Matlab.dot'],...
%       gca,'layoutengine',pathname); box on
cd('..')
end

function [Adj,Adj_net,arcdir]=stoich2adj(Stoich,S)
% Transform the stoichometric matrix to the adjoint state matrix.
% stoich_matrix could be a real number matrix or text cell matrix, in order
% to rapresent on the pathnet graph respectivly the values or the symbolic
% stoich coefficients.

% Get the sign of the stoich matrix in order to identify the reactants and
% the product of the reactions
stoich_matrix_sign=sign(Stoich);

% Initialize the adjoint state matrix
[N_states,N_rates]=size(stoich_matrix_sign);
Adj=cell(N_states,N_states);
Adj_net=zeros(N_states,N_states);
arcdir=Adj_net;
%idx_Adj=[];
for i=1:N_rates
    for j=1:N_states
        if stoich_matrix_sign(j,i)==-1
            reactant = j;
            for k=1:N_states
                if stoich_matrix_sign(k,i)==1 && k~=j
                    %idx_Adj = [idx_Adj; [reactant,k,i]];
                if iscell(Stoich)
                    if isreal(Stoich{k,i})
                        stoich=num2str(Stoich{k,i});
                    else stoich=Stoich{k,i};
                    end
                else stoich=num2str(Stoich(k,i));
                end
                    Adj(reactant,k)={[stoich ',' S.Rates_name{i}]};
                    Adj_net(reactant,k)=1;
                    arcdir(reactant,k)=S.rates_direction{i};
                end
            end
        end
    end
end

end

function graph2dot(filename,adj_val,adj,arcdir,S)
% GraphViz dot file given by a adjacency matrix.
% Some options
width=30;
height=30;

% Get the necessary inforamtion for the arcs and nodes

% Open the file to write
fid = fopen(filename, 'w');
fprintf(fid, 'digraph G {\n');

% Write the dot file general options
fprintf(fid,...
 ['graph [rotate=0]\n',...
  'node  [shape=ellipse, fontname="Trebuchet MS", fontsize="10"]\n',...
  'edge  [color="#666666",fontname="Trebuchet MS", fontsize="8"]\n']);
fprintf(fid, 'center = 1;\n');
fprintf(fid, 'size=\"%d,%d\";\n', width, height);
fprintf(fid, 'rankdir=TB;\n');

% Process the nodes
Nnds = length(adj);
for node = 1:Nnds
    str='%d [ label = "%s",style=filled,color="#%s"];\n';
    fprintf(fid,str,node,S.states_graph{node,1},S.states_graph{node,2});
end

% Process the subgraphs
Gstr=grouprank_string(grouprank(S));
for sub=1:length(Gstr)
    fprintf(fid,[Gstr{sub} '\n']);
end

% Process the arcs
for node1 = 1:Nnds
    arcs = find(adj(node1,:));  % children(adj, node);
    stropt=['fontcolor="#'  S.states_graph{node1,2} '"'];
    for node2 = arcs
        if arcdir(node1,node2)==0 % unidirected arc
            strarc=[num2str(node1) '->' num2str(node2) ' [label="' adj_val↙
{node1,node2} '",' stropt ',color="#' S.states_graph{node1,2} '"];\n'];
        else % Bidirected arc
            strbi=['dir=both,color="#' S.states_graph{node1,2} ':#' S.↙
states_graph{node2,2} '"];\n'];
            strarc=[num2str(node1) '->' num2str(node2) ' [label="' adj_val↙
{node1,node2} '",' stropt ',' strbi ];
        end
        fprintf(fid,strarc);
    end
end
% Close the file
fprintf(fid, '\n}');
fclose(fid);
end
```

```
function Code=buildCode(S,Codebasic)
% Build the code for symbolic, C/C++ and Matlab

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Define the variables to use for the parameters, the rates and the
% functions declarations. States, derivatives and I/O are fixed since
% simulink use a default nomenclature
var={'p','r','f'};
Code=Codebasic;

% Symbolic, C and Matlab defined variables
varStates=var2idx(S.States_name,'x');
varDeriv=var2idx(S.States_name,'dx');
varInput=var2idx(S.Input_name(2:end),'u');
varOutput=var2idx(S.Output_name,'y');
varPar=var2idx(S.Parameters_name(1:end-(size(S.States_name(:),1))+1),...
    var{1});
varRates=var2idx(S.Rates_name,var{2});
varFunc=var2idx(S.Functions_name,var{3});
Code.varStates=varStates;
Code.varDeriv=varDeriv;
Code.varInput=varInput;
Code.varOutput=varOutput;
Code.var=var;
Code.Mass=S.xls.Mass;
Code.Drates=cell2mat(S.xls.rates(:,4));

% Find the states that are function of differentials
[r,c] = size(S.xls.stoich); A=zeros(r,c);
for i=1:r
    for j=1:c
        if isfloat(S.xls.stoich{i,j})
            A(i,j)=(S.xls.stoich{i,j}==0);
        elseif ischar(S.xls.stoich{i,j})
            A(i,j)=0;
        end
    end
end
if any(Code.Drates,1)
    Code.Dstates = ~A(:,logical(Code.Drates));
else
    Code.Dstates = zeros(r,1);
end

% Symbolic, C and Matlab defined equations
sysvar=addspace([varStates;varPar;varInput;varFunc;varRates;varOutput]);
```

```
C_rates_equations=sym2ode(addspace(S.rates_equations),sysvar);
for i = 1:size(Code.Drates,1) % Raplace the states with the differentials
    if Code.Drates(i)==true
        C_rates_equations(i,2)=...
            regexprep(C_rates_equations(i,2),'x','dx','matchcase');
        C_rates_equations(i,3)=...
            regexprep(C_rates_equations(i,3),'x','dx','matchcase');
    end
end
C_func_equations=sym2ode(addspace(S.func_equations),sysvar);
C_ODE_equations=sym2ode(addspace(S.ODE_equations),sysvar);
C_output_equations=sym2ode(addspace(S.output_equations),sysvar);

% Jacobian matrix for Matlab
try
    [n,m]=size(S.Jacobian);
    M_jacobian_matrix=sym2ode(reshape(addspace(S.Jacobian),1,m*n),sysvar);
    Code.Jacobian=reshape(M_jacobian_matrix,n,m,3);
end

% Control that there are no improper names
flag=controlname(sysvar);

% Couple the variables with the corresponding equations
Code.N_states=size(varStates,1);
Code.N_param =size(varPar,1);
Code.N_input =size(varInput,1);
Code.N_output=size(varOutput,1);
Code.N_func  =size(varFunc,1);
Code.N_rates =size(varRates,1);
Code.Func =cell(Code.N_func,3);
Code.Rates=cell(Code.N_rates,3);
Code.Deriv=cell(Code.N_states,3);
Code.Output=cell(Code.N_output,3);

%%%%%%%%% Variables and Parameters in C
Code.CdeclareVar = {['double *' var{1} ];...
    ['double ' var{2} '[' num2str(Code.N_rates) ']=0'];...
    ['double ' var{3} '[' num2str(Code.N_func)  ']=0']};

%%%%%%%%% Funtions
for i=1:Code.N_func
    for j=1:3
        Code.Func{i,j} =[varFunc{i,j}  ' = ' C_func_equations{i,j}];
    end
end
%%%%%%%%% Rates
for i=1:Code.N_rates
    for j=1:3
        Code.Rates{i,j}=[varRates{i,j} ' = ' C_rates_equations{i,j}];
    end
end
%%%%%%%%% Derivatives
for i=1:Code.N_states
    for j=1:3
        Code.Deriv{i,j}=[varDeriv{i,j} ' = ' C_ODE_equations{i,j}];
    end
end
%%%%%%%%% Outputs
for i=1:Code.N_output
    for j=1:3
        Code.Output{i,j}=[varOutput{i,j} ' = ' C_output_equations{i,j}];
    end
end
```

```
function CmexModel(Code,compile)
% Write the Cmex-code and compile it.

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

% Store the generated files in the Models subdirectory
cd( [cd '/Models'])

% Open the txt document
fid = fopen([Code.modelname '.c'], 'w');

% Author and model general description (notes)
for i=1:length(Code.infoauthor)
    fprintf(fid, ['\n // ' Code.infoauthor{i}] );
end
fprintf(fid,'\n');

for i=1:length(Code.infomodel)
    fprintf(fid, ['\n // Notes:']);
    fprintf(fid, ['\n // ' Code.infomodel{i} ] );
end
fprintf(fid,'\n');

% Definitions for the Cmex and the libraries used
str=['\n #define S_FUNCTION_NAME ' Code.modelname,...
    '\n #include "simstruc.h" ',...
    '\n #include <math.h>',...
    '\n #define InitialConditions     ssGetSFcnParam(S,0)',...
    '\n #define Parameters        ssGetSFcnParam(S,1)'];
fprintf(fid, [str, '\n']);

% Initialize the size of the SimStuct
str=['\n static void mdlInitializeSizes(SimStruct *S)',...
    '\n{',...
    '\n ssSetNumContStates(    S, ' num2str(Code.N_states) ');',...
    '\n ssSetNumDiscStates(    S, 0);',...
    '\n ssSetNumInputs(        S, ' num2str(Code.N_input) ');',...
    '\n ssSetNumOutputs(       S, ' num2str(Code.N_output) ');',...
    '\n ssSetDirectFeedThrough(S, 1);',...
    '\n ssSetNumSampleTimes(   S, 1);',...
    '\n ssSetNumSFcnParams(    S, 2);',...
    '\n ssSetNumRWork(         S, 0);',...
    '\n ssSetNumIWork(         S, 0);',...
    '\n ssSetNumPWork(         S, 0);',...
    '\n }'];
fprintf(fid, [str, '\n']);
```

```
% Define the sample time
str=['\n static void mdlInitializeSampleTimes(SimStruct *S)',...
    '\n {',...
    '\n ssSetSampleTime(S, 0, CONTINUOUS_SAMPLE_TIME);',...
    '\n ssSetOffsetTime(S, 0, 0.0);',...
    '\n }'];
fprintf(fid, [str, '\n']);

% Define the initial conditions of the states
str=['\n static void mdlInitializeConditions(double *x0, SimStruct *S)',...
    '\n {',...
    '\n int i;',...
    '\n for (i=0; i<' num2str(Code.N_states) '; i++) {x0[i]=mxGetPr↙
(InitialConditions)[i];}',...
    '\n }'];
fprintf(fid, [str, '\n']);

%%%%%%%%%%% Outputs

str=['\n static void mdlOutputs(double *y, double *x, double *u, SimStruct↙
*S, int tid)',...
    '\n {',...
    '\n // outputs',...
    '\n // Declare variables'];
fprintf(fid, str);

for i=1:size(Code.CdeclareVar,1)
    fprintf(fid, ['\n ' Code.CdeclareVar{i} ';'] );
end
fprintf(fid,'\n');

str=['\n // Parameters',...
    '\n p=mxGetPr(Parameters);',...
    '\n\n // Functions'];
fprintf(fid, str);

for i=1:size(Code.Func,1)
    fprintf(fid, ['\n ' Code.Func{i,2} ';'] );
end
fprintf(fid,'\n');

for i=1:size(Code.Output,1)
    fprintf(fid, ['\n ' Code.Output{i,2} ';'] );
end
fprintf(fid,'\n }');

%%%%%%%%%% Update
str=['\n\n static void mdlUpdate(double *x, double *u, SimStruct *S, int↙
tid)',...
    '\n {',...
    '\n }'];
fprintf(fid, str);

%%%%%%%%%%% Continous states
str=['\n\n static void mdlDerivatives(double *dx, double *x, double *u,↙
SimStruct *S, int tid)',...
    '\n {',...
    '\n // Declare variables'];
fprintf(fid, str);

for i=1:size(Code.CdeclareVar,1)
    fprintf(fid, ['\n ' Code.CdeclareVar{i} ';'] );
end
fprintf(fid,'\n');

str=['\n // Parameters',...
    '\n p=mxGetPr(Parameters);',...
    '\n\n // Functions'];
fprintf(fid, str);

for i=1:size(Code.Func,1)
    fprintf(fid, ['\n ' Code.Func{i,2} ';'] );
end
fprintf(fid,'\n');

str=['\n // Rates']; fprintf(fid, str);
for i=1:size(Code.Rates,1)
    if Code.Drates(i)==false
        fprintf(fid, ['\n ' Code.Rates{i,2} ';'] );
    end
end
fprintf(fid,'\n');

str=['\n // Derivatives']; fprintf(fid, str);
for i=1:size(Code.Deriv,1)
    if Code.Dstates(i)==false
        fprintf(fid, ['\n ' Code.Deriv{i,2} ';'] );
    end
end
fprintf(fid,'\n');

str=['\n // Differential Rates']; fprintf(fid, str);
for i=1:size(Code.Rates,1)
    if Code.Drates(i)==true
        fprintf(fid, ['\n ' Code.Rates{i,2} ';'] );
    end
end
fprintf(fid,'\n');

str=['\n // Derivatives']; fprintf(fid, str);
```

```
for i=1:size(Code.Deriv,1)
    if Code.Dstates(i)==true
        fprintf(fid, ['\n ' Code.Deriv{i,2} ';'] );
    end
end
fprintf(fid,'\n }');


%%%%%%%%%% End mex statement
str=['\n\n static void mdlTerminate(SimStruct *S)',...
    '\n {',...
    '\n }',...
    '\n\n #ifdef    MATLAB_MEX_FILE',...
    '\n #include "simulink.c"',...
    '\n #else',...
    '\n #include "cg_sfun.h"',...
    '\n #endif'];
fprintf(fid, str);

% Close the written file
fclose(fid);

if compile==true
    eval(['mex ' Code.modelname '.c']);
end

% Return to the current directory
cd( '..')
```

```
function MfileModel(Code,type,Mass_matrix)

% Store the generated files in the Models subdirectory
cd( [cd '/Models'])

% Open the txt document
fid = fopen([Code.modelname '_f.m'], 'w');

% Author and model general description (notes)
for i=1:length(Code.infoauthor)
    fprintf(fid, ['\n %% ' Code.infoauthor{i}] );
end
fprintf(fid,'\n');

fprintf(fid, '\n %% Notes:' );
for i=1:length(Code.infomodel)
    fprintf(fid, ['\n %% ' Code.infomodel{i} ] );
end
fprintf(fid,'\n');

% Define the DAE-ODE system
switch type

    case 'sundials'
        n=size(Code.Deriv,1);
        res=cell(n,1);
        for i=1:n
            if Mass_matrix(i)==true
                res{i}=['-p'  Code.varStates{i,3} ];
            else
                res{i}=' ';
            end
        end

        if ~any(Mass_matrix)
            str=['\n function [res, flag, new_data] = ' Code.modelname '_f(t,↙
x,Data)\n'];
        elseif ~all(Mass_matrix)
            str=['\n function [res, flag, new_data] = ' Code.modelname '_f(t,↙
x,px,Data)\n'];
        elseif all(Mass_matrix)
            str=['\n function [dx, flag, new_data] = ' Code.modelname '_f(x,↙
Data)\n'];
        end
        fprintf(fid, str);
        % Define the size of the memory
        fprintf(fid, '\n\n %%Allocate arrays' );
        fprintf(fid,'\n dx=zeros(Data.AllocateMemory(1),1);');
        fprintf(fid,'\n f=zeros(Data.AllocateMemory(2),1);');
        fprintf(fid,'\n r=zeros(Data.AllocateMemory(3),1);');
```

```
%fprintf(fid,'\n y=zeros(Data.AllocateMemory(4),1);');

% Define the parameters
fprintf(fid, '\n\n %%Parameters' );
fprintf(fid,['\n ' Code.var{1} '=Data.Parameters;']);
% Define the inflow
fprintf(fid, '\n\n %%Inflow' );
if ~any(Mass_matrix) || ~all(Mass_matrix)
    fprintf(fid,'\n siminTime=Data.simin(:,1);');
    fprintf(fid,'\n simin=Data.simin(:,2:end);');
    fprintf(fid,'\n u=interp1q(siminTime,simin,t);');
elseif all(Mass_matrix)
    fprintf(fid,'\n u=Data.simin(1,2:end);');
end
% Print the functions
fprintf(fid, '\n\n %%Functions' );
for i=1:size(Code.Func,1)
    fprintf(fid, ['\n ' Code.Func{i,3} ';'] );
end
fprintf(fid,'\n');
% Print the process rates
fprintf(fid, '\n %%Rates' );
for i=1:size(Code.Rates,1)
    fprintf(fid, ['\n ' Code.Rates{i,3} ';'] );
end
fprintf(fid,'\n');
% Print the derivatives
fprintf(fid, '\n %%Derivatives' );
for i=1:size(Code.Deriv,1)
    fprintf(fid, ['\n ' Code.Deriv{i,3} ';'] );
end
fprintf(fid,'\n');
if ~any(Mass_matrix) || ~all(Mass_matrix)
    % Print the residuals
    fprintf(fid, '\n %%Residuals' );
    fprintf(fid, '\n res=[' );
    for i=1:size(Code.Deriv,1)
        fprintf(fid, ['\n ' res{i} ' + ' Code.varDeriv{i,3} ';'] );
    end
    fprintf(fid,'\n ];\n');
end

%         % Print the outputs
%         fprintf(fid, '\n %%Outputs' );
%         for i=1:size(Code.Output,1)
%             fprintf(fid, ['\n ' Code.Output{i,3} ';'] );
%         end
%         fprintf(fid,'\n y=y(:);');

fprintf(fid, '\n flag = 0; \n new_data = [];');
```

```
    case 'ode15s'
        error('ode15s not implemented jet!')

    otherwise
        error('Sundials or Ode15s solvers are only available.')
end

% Close the written file
fclose(fid);



% Open the txt document for the Jacobian
fid = fopen([Code.modelname '_J.m'], 'w');

% Author and model general description (notes)
for i=1:length(Code.infoauthor)
    fprintf(fid, ['\n %% ' Code.infoauthor{i}] );
end
fprintf(fid,'\n');

fprintf(fid, '\n %% Notes:' );
for i=1:length(Code.infomodel)
    fprintf(fid, ['\n %% ' Code.infomodel{i} ' - Jacobian matrix'] );
end
fprintf(fid,'\n');

switch type

    case 'sundials'
        % Print the Jacobian matrix
        if ~any(Mass_matrix)
            str=['\n function [J, flag, new_data] = ' Code.modelname '_J(t,x,↙
px,Data)\n'];
        else
            str=['\n function [J, flag, new_data] = ' Code.modelname '_J(t,x,↙
px,rr,cj,Data)\n'];
        end
        fprintf(fid, str);

        % Define the parameters
        fprintf(fid, '\n\n %%Parameters' );
        fprintf(fid,['\n ' Code.var{1} '=Data.Parameters;']);
        % Define the inflow
        fprintf(fid,'\n\n %%Inflow' );
        fprintf(fid,'\n siminTime=Data.simin(:,1);');
        fprintf(fid,'\n simin=Data.simin(:,2:end);');
        fprintf(fid,'\n u=interp1q(siminTime,simin,t);');
        % Print the functions
        fprintf(fid, '\n\n %%Functions' );
        for i=1:size(Code.Func,1)
            fprintf(fid, ['\n ' Code.Func{i,3} ';'] );
        end
        % Define the Jacobian matrix
        fprintf(fid,'\n\n %%Jacobian matrix' );
        for i=1:size(Code.Jacobian(:,:,3),1)
            for j=1:size(Code.Jacobian(:,:,3),2)
                if i==j && Mass_matrix(j) ==1
                    strcj='- cj ;';
                else
                    strcj=' ;';
                end
                fprintf(fid, ['\n J(' num2str(i) ',' num2str(j) ') = ' Code.↙
Jacobian{i,j,3} strcj]);
            end
        end
        fprintf(fid,'\n');

        fprintf(fid, '\n flag = 0; \n new_data = [];');


    case 'ode15s'


    otherwise
        error('Sundials or Ode15s solvers are only available.')
end


% Close the written file
fclose(fid);


% Return to the current directory
cd('..')
```

```
function PNM=modelclass(S,Code,scaletype)
% Path Network Model class. All the data needed for simulation.

% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

if nargin<3
    % All the measurments are weighted the same
    scaletype='one';
end

% Multi-batch input flow
In     =cell2mat(S.xls.input(3:end,2:end));
nbatch =cell2mat(S.xls.input(3:end,1));
Inflow =cell(max(nbatch),1);

% Data organization
data=opendatastruct(S.xls.dataxls);
for i=1:length(data)
    Inflow{i}=In(i==nbatch,:);
end

% Build the class
PNM            =PNMclass;
PNM.Code       =Code;
PNM.Data       =data;
PNM.Measures   =S.Data_name;
PNM.Parameters =S.param_mcmc;
PNM.Inflow     =Inflow;
PNM.Output.Output_names =S.Output_name;
PNM.Output.Output_units =S.output_units;
PNM.Scale               =scaleval(PNM,scaletype);

end

function S=scaleval(PNM,type)

% Number of batch experiments
nbatch =length(PNM.Data);
S      =cell(nbatch,1);

for ibatch = 1:nbatch
    switch type
        case 'one' % No scaling
            S{ibatch}=...
     ones(size(catmultidim(1,PNM.Data{ibatch}.ydata(:,2:end,:)),2),1);
        case 'max' % Scale acording to the maximum value
            S{ibatch}=...
```

```
                nanmax(catmultidim(1,PNM.Data{ibatch}.ydata(:,2:end,:)));
        case 'std' % Use STD as the normalizing constant
            S{ibatch}=...
                nanstd(catmultidim(1,PNM.Data{ibatch}.ydata(:,2:end,:)));
        otherwise
            error('Should specify the type of data scale paramater used.')
    end
end
if nbatch==1
    S=S{:};
    return
end

switch type
    case 'one' % No scaling
        S=ones(1,size(PNM.Data{1}.ydata(:,2:end,:),2));
    case 'max' % Scale acording to the maximum value
        S=nanmax(cell2mat(S));
    case 'std' % Use STD as the normalizing constant
        S=nanstd(cell2mat(S));
    otherwise
        error('Should specify the type of data scale paramater used.')
end

if sum(S<=0)
    error('The scale value could not be zero or negative.')
end

end
```

```
% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

classdef PNMclass
    % Define the PNM class
   properties
      Code;
      Data;
      Measures;
      Parameters;
      Inflow;
      Output;
      Scale;
      ModelMCMC;
      OptionsMCMC;
   end
end
```

```
function [names,value,parind,local,upper,lower,thetamu,thetasig,hpar] = ...
    openparstruct(parstruct,nbatch)
%OPENPARSTRUCT parameter struct utility for mcmc tbx
% [names,value,parind,local,upper,lower,thetamu,thetasig] = ...
%    openparstruct(parstruct,nbatch)

% $Revision: 1.8 $  $Date: 2009/08/22 21:25:35 $

ntheta    = length(parstruct);
local     = [];
npar      = ntheta;

% parstruct has
%    1       2      3    4   5    6     7       8
% {'name', theta0, min, max, mu, sig, sample, local}

% for hyperpriors, local = 2
%    1       2      3    4      5         6       7       8
% {'name', theta0, min, max, [mu,tau], [sig,n], sample, local}
% and set up hpar
% hpar.ind, hpar.mu0, hpar.tau20, hpar.sig20, hpar.n0

ii = 0; nhpar = 0;
%% scan for local variables
for i=1:length(parstruct)
  ii = ii+1;
  local(ii) = 0;
  if length(parstruct{i})>7
    if parstruct{i}{8}
      if parstruct{i}{8}==2
    nhpar=nhpar+1;
      end
      local(ii:(ii+nbatch-1)) = 1:nbatch;
%     ntheta=ntheta+nbatch-1;
      npar=npar+nbatch-1;
      ii = ii+nbatch-1;
      for k=2:7
    if parstruct{i}{8}==2 & (k==5|k==6)
      if not(length(parstruct{i}{k})==1|length(parstruct{i}{k})==2)
        error(sprintf('Error in hyper parameters for %s',parstruct{i}{1}))
      end
    else
      if length(parstruct{i}{k})~=nbatch
        if length(parstruct{i}{k})==1
          parstruct{i}{k} = parstruct{i}{k}*ones(1,nbatch);
        else
          error(sprintf('Not enough values for %s',parstruct{i}{1}))
        end
      end
    end
```

```
      end
    end
  end
end

%local

value     = zeros(npar,1);
names     = cell(npar,1);
upper     = ones(1,npar)*Inf;
lower     = -ones(1,npar)*Inf;
thetamu   = zeros(1,npar);
thetasig  = ones(1,npar)*Inf;
parind    = ones(npar,1);

hpar.ind = zeros(1,npar);
hpar.mu0 = zeros(1,nhpar);
hpar.tau20 = zeros(1,nhpar);
hpar.sig20 = zeros(1,nhpar);
hpar.n0 = zeros(1,nhpar);
hpar.names = {};

ii = 0; ihpar = 1;
for i=1:ntheta
  ii = ii+1;
  %  assignin('base',parstruct{i}{1},parstruct{i}{2});
  if local(ii) == 0
    names{ii}   = parstruct{i}{1};
    value(ii)   = parstruct{i}{2};

    if length(parstruct{i})>2 & ~isempty(parstruct{i}{3})
      lower(ii)    = parstruct{i}{3};
    end
    if length(parstruct{i})>3 & ~isempty(parstruct{i}{4})
      upper(ii)    = parstruct{i}{4};
    end
    if length(parstruct{i})>=6
      thetamu(ii)  = parstruct{i}{5};
      thetasig(ii) = parstruct{i}{6};
      if isnan(thetamu(ii))
        thetamu(ii)=value(ii);
      end
      if thetasig(ii) == 0
        thetasig(ii) = Inf;
      end
    end
    if length(parstruct{i})>=7 & parstruct{i}{7}==0% parind-flagi
      parind(ii) = 0;
    end
  else

    iii = ii:(ii+nbatch-1);
    if nbatch==1
        names(iii(1))    = {sprintf('%s',parstruct{i}{1})};
    else
      for k=1:nbatch
        names(iii(k))    = {sprintf('%s[%d]',parstruct{i}{1},k)};
      end
    end
    value(iii)    = parstruct{i}{2};

    if length(parstruct{i})>2 & ~isempty(parstruct{i}{3})
      lower(iii)    = parstruct{i}{3};
    end
    if length(parstruct{i})>3 &~isempty(parstruct{i}{4})
      upper(iii)    = parstruct{i}{4};
    end
    if length(parstruct{i})>=6
      if length(parstruct{i})>=8 & parstruct{i}{8}==2 % hyperprior
    if isnan(parstruct{i}{5}(1))
      hpar.mu0(ihpar) = parstruct{i}{2}(1);
    else
      hpar.mu0(ihpar) = parstruct{i}{5}(1);
    end
        if length(parstruct{i}{5})>1;
          hpar.tau20(ihpar) = parstruct{i}{5}(2)^2;
        else
          hpar.tau20(ihpar) = Inf;
        end
    hpar.sig20(ihpar) = parstruct{i}{6}(1)^2;
        if length(parstruct{i}{6})>1;
          hpar.n0(ihpar) = parstruct{i}{6}(2);
        else
          hpar.n0(ihpar) = 0;
        end
    hpar.ind(iii) = ihpar;
    hpar.names = {hpar.names{:},sprintf('mu(%s)',parstruct{i}{1}),sprintf↙
('sig(%s)',parstruct{i}{1})};
    % initial values as mu0 and sig20
    thetamu(iii)  = hpar.mu0(ihpar);
    thetasig(iii) = sqrt(hpar.sig20(ihpar)); %%%
    ihpar = ihpar+1;
      else
      thetamu(iii)  = parstruct{i}{5};
      thetasig(iii) = parstruct{i}{6};
      for i2=iii
        if isnan(thetamu(i2))
          thetamu(i2)=value(i2);
        end
        if thetasig(i2) == 0
          thetasig(i2) = Inf;
        end
      end
      end
    end
    if length(parstruct{i})>=7 & parstruct{i}{7}==0% parind-flagi
      parind(iii) = 0;
    end

    ii = ii + nbatch - 1 ;

  end

end
parind = find(parind);

hpar.ind = hpar.ind(parind);
hpar.nhpar = nhpar;

%local=0; %%%%%% not implemented yet
```

```
function ha = tight_subplot(Nh, Nw, gap, marg_h, marg_w)

% tight_subplot creates "subplot" axes with adjustable gaps and margins
%
% ha = tight_subplot(Nh, Nw, gap, marg_h, marg_w)
%
%   in:  Nh      number of axes in hight (vertical direction)
%        Nw      number of axes in width (horizontaldirection)
%        gap     gaps between the axes in normalized units (0...1)
%                   or [gap_h gap_w] for different gaps in height and width
%        marg_h  margins in height in normalized units (0...1)
%                   or [lower upper] for different lower and upper margins
%        marg_w  margins in width in normalized units (0...1)
%                   or [left right] for different left and right margins
%
%  out:  ha     array of handles of the axes objects
%                   starting from upper left corner, going row-wise as in
%                   going row-wise as in
%
%  Example: ha = tight_subplot(3,2,[.01 .03],[.1 .01],[.01 .01])
%           for ii = 1:6; axes(ha(ii)); plot(randn(10,ii)); end
%           set(ha(1:4),'XTickLabel',''); set(ha,'YTickLabel','')

% Pekka Kumpulainen 20.6.2010   @tut.fi
% Tampere University of Technology / Automation Science and Engineering


if nargin<3; gap = .02; end
if nargin<4 || isempty(marg_h); marg_h = .05; end
if nargin<5; marg_w = .05; end

if numel(gap)==1;
    gap = [gap gap];
end
if numel(marg_w)==1;
    marg_w = [marg_w marg_w];
end
if numel(marg_h)==1;
    marg_h = [marg_h marg_h];
end

axh = (1-sum(marg_h)-(Nh-1)*gap(1))/Nh;
axw = (1-sum(marg_w)-(Nw-1)*gap(2))/Nw;

py = 1-marg_h(2)-axh;

ha = zeros(Nh*Nw,1);
ii = 0;
for ih = 1:Nh
    px = marg_w(1);
    for ix = 1:Nw
        ii = ii+1;
        ha(ii) = axes('Units','normalized', ...
            'Position',[px py axw axh], ...
            'XTickLabel','', ...
            'YTickLabel','');
        px = px+axw+gap(2);
    end
    py = py-axh-gap(1);
end
```

```
function [H, pValue, KSstatistic] = kstest_2s_2d(x1, x2, alpha, tail)

%
% Two-sample Two-diensional Kolmogorov-Smirnov Test
%
% modified from MATLAB built-in function "kstest2" from Statistics Toolbox
% usage: same as "kstest2"
%
% algorithm summary (Peak, 1983): consider the four quadrants (x<X,y<Y),
% (x<X,y>Y), (x>X,y<Y) and (x>X,y>Y) in turn, and adopt the largest of the
% four differences between the two empirical cumulative distributions as
% the final KS statistic.
%
% Author: Qiuyan Peng @ ECE/HKUST
% Date: 11th May, 2011
%
% References:
% J. A. Peacock, "Two-dimensional goodness-of-fit testing in astronomy",
%   Monthly Notices Royal Astronomy Society 202 (1983) 615-627.
%


%%

if nargin < 2
    error('stats:kstest2:TooFewInputs','At least 2 inputs are required.');
end


%%
%
% x1,x2 are both 2-column matrices
%

if ((size(x1,2)~=2)||(size(x2,2)~=2))
    error('stats:kstest2:TwoColumnMatrixRequired','The samples X1 and X2 must↙
be two-column matrices.');
end
n1 = size(x1,1);
n2 = size(x2,1);


%%
%
% Ensure the significance level, ALPHA, is a scalar
% between 0 and 1 and set default if necessary.
%

if (nargin >= 3) && ~isempty(alpha)
   if ~isscalar(alpha) || (alpha <= 0 || alpha >= 1)
```

```
      error('stats:kstest2:BadAlpha',...
            'Significance level ALPHA must be a scalar between 0 and 1.');
   end
else
   alpha  =  0.05;
end


%%
%
% Ensure the type-of-test indicator, TYPE, is a scalar integer from
% the allowable set, and set default if necessary.
%

if (nargin >= 4) && ~isempty(tail)
   if ischar(tail)
      tail = strmatch(lower(tail), {'smaller','unequal','larger'}) - 2;
      if isempty(tail)
         error('stats:kstest2:BadTail',...
               'Type-of-test indicator TYPE must be ''unequal'', ''smaller'',↙
or ''larger''.');
      end
   elseif ~isscalar(tail) || ~((tail==-1) || (tail==0) || (tail==1))
      error('stats:kstest2:BadTail',...
            'Type-of-test indicator TYPE must be ''unequal'', ''smaller'', or↙
''larger''.');
   end
else
   tail  =  0;
end


%%
%
% Calculate F1(x) and F2(x), the empirical (i.e., sample) CDFs.
%

tOp = {'>=','>='; '<=','>='; '<=','<='; '>=','<='};
nOp = 4;
deltaCDF = zeros(nOp,n1+n2);

for iX = 1:(n1+n2)
    if (iX<=n1)
        edge = x1(iX,:);
    else
        edge = x2(iX-n1,:);
    end

    for iOp = 1:nOp
```

```
        eval(['sel_1 = (x1(:,1)',tOp{iOp,1},'edge(1))&(x1(:,2)',tOp{iOp,↙
2},'edge(2));']);
        eval(['sel_2 = (x2(:,1)',tOp{iOp,1},'edge(1))&(x2(:,2)',tOp{iOp,↙
2},'edge(2));']);
        sampleCDF1 = sum(sel_1)/n1;
        sampleCDF2 = sum(sel_2)/n2;

        switch tail
            case  0      %  2-sided test: T = max|F1(x) - F2(x)|.
                deltaCDF(iOp,iX)  =  abs(sampleCDF1 - sampleCDF2);
            case -1      %  1-sided test: T = max[F2(x) - F1(x)].
                deltaCDF(iOp,iX)  =  sampleCDF2 - sampleCDF1;
            case  1      %  1-sided test: T = max[F1(x) - F2(x)].
                deltaCDF(iOp,iX)  =  sampleCDF1 - sampleCDF2;
        end

    end

end

KSstatistic   =  max(deltaCDF(:));


%%
%
% Compute the asymptotic P-value approximation and accept or
% reject the null hypothesis on the basis of the P-value.
%

n      =  n1 * n2 /(n1 + n2);
lambda =  max((sqrt(n) + 0.12 + 0.11/sqrt(n)) * KSstatistic , 0);

if tail ~= 0        % 1-sided test.

   pValue  =  exp(-2 * lambda * lambda);

else                % 2-sided test (default).
%
%  Use the asymptotic Q-function to approximate the 2-sided P-value.
%
   j       =  (1:101)';
   pValue  =  2 * sum((-1).^(j-1).*exp(-2*lambda*lambda*j.^2));
   pValue  =  min(max(pValue, 0), 1);

end

H  =  (alpha >= pValue);
```

```
function varargout=plims2d(xy,lims,smo,rho,xo,yo)
%PLIMS2D  2 dimensional HPD limits
% Calculated 2d highest posterior density probability limits.
% This is used by PANELLIMS.

% Marko Laine <Marko.Laine@Helsinki.FI>
% $Revision: 1.4 $  $Date: 2008/01/23 08:02:53 $

if nargin < 3
   smo = 1;
end
if nargin < 4
   rho = [];
end
if nargin < 5
   xo=[];
end
if nargin < 6
   yo=[];
end

[z,xo,yo]=density2d(xy,xo,yo,smo,rho);

%c=cumsum(sort(z(:).*diff(xo).*diff(yo)));

% locate the confidence regions
 d  = (xo(end)-xo(end-1))*(yo(end)-yo(end-1));
 zs = sort(z(:));
 g  = zs*d;
 cumu = cumsum(g);
 %disp(sprintf('Total mass: %g\n',cumu(length(cumu))))

 sc = zeros(length(lims),1);
 for j=1:length(lims)
      i = find(cumu<(1-lims(j)));
      sc(j) = zs(length(i));
 end

 if nargout==1
    varargout{1}=sc;
 elseif nargout==4
    varargout{1}=xo;
    varargout{2}=yo;
    varargout{3}=z;
    varargout{4}=sc;
 end
```

```
function [z,xo,yo,s]=density2d(x,xout,yout,ss,rho,plotit)
%DENSITY2D   2 dimensinal density estimator
% [z,x,y]=density2d(x,xout,yout,s,rho,plotit)
% x size n*2 data used for estimation
% xout  1. coordinate points returned (optional)
% yout  2. coordinate points returned (optional)
% s relative smoothing factor (default = 1)
% rho correlation coefficient of the Gaussian kernel used (optional)
% plotit 0 = no plot, 1 = contour plot, 2 = mesh plot
%
% output: z,x,y cordinates of the estimator

% ML 2000, see MASS 2nd ed, page 184

% Marko Laine <Marko.Laine@Helsinki.FI>
% $Revision: 1.5 $  $Date: 2008/01/23 08:02:52 $

numpoints=50;

if size(x,2)~= 2
   error('size(x,2)~=2')
end
nx=size(x,1);
if nargin<2 | isempty(xout)
   xmin=min(x(:,1)); xmax=max(x(:,1)); xrange=xmax-xmin;
   xout=linspace(xmin-0.06*xrange,xmax+0.06*xrange,numpoints);
end
if nargin<3 | isempty(yout)
   xmin=min(x(:,2)); xmax=max(x(:,2)); xrange=xmax-xmin;
   yout=linspace(xmin-0.06*xrange,xmax+0.06*xrange,numpoints);
end
s1=1.06*min(std(x(:,1)),iqrange(x(:,1))/1.34)*nx^(-1/6); % -1/5
s2=1.06*min(std(x(:,2)),iqrange(x(:,2))/1.34)*nx^(-1/6); %
%% fixme, when iqrange = 0
if s1 == 0
  s1 = 1.06*std(x(:,1))*nx^(-1/6);
end
if s2 == 0
  s2 = 1.06*std(x(:,2))*nx^(-1/6);
end
s=[s1,s2];
if nargin>3 & ~isempty(ss)
   s=s.*ss;
end
if nargin<5 | isempty(rho)
% rho=0;
  rho4=corrcoef(x); rho=rho4(1,2);
else
  if abs(rho)>=1
    error('rho should be between -1 and 1')
  end
end
if nargin<6
   plotit=0;
end

[X,Y]=meshgrid(xout,yout);

[mX,nX]=size(X);
z=zeros(mX,nX);

r = 1-rho.^2;
c = 1./(2*pi*s(1)*s(2)*sqrt(r));
for i=1:(mX*nX)
  if 0
   z(i)=1/nx*sum(norpf((X(i)-x(:,1))/s(1)).*...
      norpf((Y(i)-x(:,2))/s(2)))/prod(s);
  else
   z(i) = 1./nx .* sum(c * exp(-0.5/r*( ...
          ((X(i)-x(:,1))./s(1)).^2 - ...
       2*rho*(X(i)-x(:,1))./s(1).*(Y(i)-x(:,2))./s(2) + ...
          ((Y(i)-x(:,2))./s(2)).^2 )));
  end
end

xout(xout<0)=0;
yout(yout<0)=0;

if nargout>1
   xo=xout;
end
if nargout>2
   yo=yout;
end

if plotit==1 | nargout == 0
   h=plot(x(:,1),x(:,2),'.','MarkerSize',4,'Color',[.5 .5 .5]);
   hold on;
   contour(xout,yout,z,10);
   hold off
elseif plotit==2
   mesh(xout,yout,z);
%   hold on;
%   plot(x(:,1),x(:,2),'o');
%   hold off
end
```

```
function y=norpf(x,mu,sigma2)
% NORPF(x,mu,sigma2)  Normal (Gaussian) density function

% Marko Laine <Marko.Laine@Helsinki.FI>
% $Revision: 1.3 $  $Date: 2007/05/21 10:37:10 $

if nargin < 2, mu=0; end
if nargin < 3, sigma2=1; end
y=1./sqrt(2*pi*sigma2).*exp(-0.5*(x-mu).^2 ./sigma2);
```

```
function [y,xo]=density(x,xout,ss,gaus)
%DENSITY  Density estimator using Gaussian kernel
% Y = DENSITY(X,XOUT,S)
% X is the vector of data values.
% The density estimator is evaluated at XOUT points.
% S is a scale factor for the default kernel bandwidth,
% default S = 1.
% Without output arguments the density is plotted.

% Marko Laine <Marko.Laine@Helsinki.FI>
% $Revision: 1.8 $  $Date: 2008/01/15 08:11:10 $

if nargin<3
  ss=1;
end
if nargin<4
  gaus=1;
end

if nargin<2 | isempty(xout)
  xmin=min(x); xmax=max(x); xrange=xmax-xmin;
  if length(x) > 200
    xout=linspace(xmin-0.08*xrange,xmax+0.08*xrange);
  else
    xout=linspace(mean(x)-4*std(x),mean(x)+4*std(x));
  end
end
y  = zeros(size(xout));
n  = length(xout);
nx = length(x);

%%% see MASS 2nd ed page 181.
if iqrange(x)<=0
  s=1.06*std(x)*nx^(-1/5);
else
  s=1.06*min(std(x),iqrange(x)/1.34)*nx^(-1/5);
end
%  s=1.144*std(x)*nx^(-1/5);
if ss>0
  s=ss*s;
elseif ss<0
  s = abs(ss);
end
if gaus
  % Gaussian kernel
  for i=1:n
    y(i) = 1/nx*sum(norpf((xout(i)-x)/s))./s;
  end
else
  % triangular kernel
  s=s*1.2113;
  for i=1:n
    y(i) = 1/nx*sum(max(0,1-abs(xout(i)-x)/s))./s;
  end
end

if nargout>1
  xo=xout;
end

if nargout==0
  plot(xout,y)
  clear y % no output
end
```

```
function [gf] = gfit(t,y,varargin)
% GFIT Computes goodness of fit for regression model
%
% USAGE:
%       [gf] = gfit(t,y)
%       [gf] = gfit(t,y,gFitMeasure)
%
% INPUT:
%           t:  vector of target values for regression model
%           y:  vector of output from regression model.
%   gFitMeasure:  string value representing different form of goodness of fit↙
measure as follows
%               '1' - mean squarred error (mse)
%               '2' - normalised mean squarred error (nmse)
%               '3' - root mean squarred error (rmse)
%               '4' - normalised root mean squarred error (nrmse)
%               '5' - mean absolute error (mae)
%               '6' - mean  absolute relative error  (mare)
%               '7' - coefficient of correlation (r)
%               '8' - coefficient of determination (d)
%               '9' - coefficient of efficiency (e)
%
% OUTPUT:
%       gf:  goodness of fit values between model output and target
%
% EXAMPLES
%      gf = git(t,y,'3');  for root mean squarred error
%
% See also

% Copyright 2004-2005 by Durga Lal Shrestha.
% eMail: durgals@hotmail.com
% $Date: 2005/07/03
% $Revision: 1.0.0 $ $Date: 2005/07/03 $

% ***********************************************************************
%% INPUT ARGUMENTS CHECK
error(nargchk(2,3,nargin));
if ~isvector(t) || ~isvector(y)
    error('Invalid data size: input data must be vector')
end
t = t(:);
y = y(:);
n = length(t);
if n ~= length(y)
    error('Invalid data size: lenght of t and y must be same')
end
if nargin == 3
    gFitMeasure = varargin{1};
else
```

```
   gFitMeasure = '1';          % default goodness of fit as mse
end;
e = t - y;
switch gFitMeasure

case '1'                      % mean squared error
    gf = mean(e.^2);        % 0 - perfect match between output and target

case '2'                      % normalised mean squared error
    gf = mean(e.^2)/var(t); % 0 - perfect match

case '3'                      % root mean squared error
    gf = sqrt(mean(e.^2));  % 0 - perfect match

case '4'                            % normalised root mean squared error
    gf = sqrt(mean(e.^2)/var(t)); % 0 - perfect match

case '5'                      % mean absolute error
   gf = mean(abs(e));       % 0 - perfect match

case '6'                      % mean absolute relative error
   gf = mean((abs(e./t)));  % 0 - perfect match

case '7'                      % coefficient of correlation
   cf = corrcoef(t,y);      % 1 - perfect match
   gf = cf(1,2);

case '8'                      % coefficient of determination
   cf = corrcoef(t,y);
   gf = cf(1,2);
   gf = gf^2;               % 1 - perfect match

case '9'                                         % coefficient of efficiency
   gf = 1 - sum(e.^2)/sum((t - mean(t)).^2); % 1 - perfect match

otherwise
  error('Invalid goodness of fit measure: It must be one of the strings {1 2↙
3 4 5 6 7 8 9}')
end
%***************************************************************************
```

```
function [results,chain,s2chain,sschain, hchain]=mcmcrun(model,data,params,↙
options,res)
%MCMCRUN Metropolis-Hastings MCMC simulation for nonlinear Gaussian models
% properties:
%  multiple y-columns, sigma2-sampling, adaptation,
%  Gaussian prior, parameter limits, delayed rejection, dram
%
% [RESULTS,CHAIN,S2CHAIN,SSCHAIN] = MCMCRUN(MODEL,DATA,PARAMS,OPTIONS)
% MODEL   model options structure
%    model.ssfun    -2*log(likelihood) function
%    model.priorfun -2*log(pior) prior function
%    model.sigma2   initial error variance
%    model.N        total number of observations
%    model.S20      prior for sigma2
%    model.N0       prior accuracy for S20
%    model.nbatch   number of datasets
%
%     sum-of-squares function 'model.ssfun' is called as
%     ss = ssfun(par,data) or
%     ss = ssfun(par,data,local)
%     instead of ssfun, you can use model.modelfun as
%     ymodel = modelfun(data{ibatch},theta_local)
%
%     prior function is called as priorfun(par,pri_mu,pri_sig) it
%     defaults to Gaussian prior with infinite variance
%
%     The parameter sigma2 gives the variances of measured components,
%     one for each. If the default options.updatesigma = 0 (see below) is
%     used, sigma2 is fixed, as typically estimated from the fitted↙
residuals.
%     If opions.updatesigma = 1, the variances are sampled as conjugate↙
priors
%     specified by the parameters S20 and N0 of the inverse gamma
%     distribution, with the 'noninformative' defaults
%          S20 = sigma2   (as given by the user)
%          N0   = 1
%     Larger values of N0 limit the samples closer to S20
%     (see,e.g., A.Gelman et all:
%     Bayesian Data Analysis, http://www.stat.columbia.edu/~gelman/book/)
%
% DATA the data, passed directly to ssfun. The structure of DATA is given
%      by the user. Typically, it contains the measurements
%
%      data.xdata
%      data.ydata,
%
%      A possible 'time' variable must be given in the first column of
%      xdata. Note that only data.xdata is needed for model simulations.
%      In addition, DATA may include any user defined strucure needed by
%      |modelfun| or |ssfun|
%
% PARAMS  theta structure
%   {  {'par1',initial, min, max, pri_mu, pri_sig, targetflag, localflag}
%      {'par2',initial, min, max, pri_mu, pri_sig, targetflag, localflag}
%      ... }
%
%   'name' and initial are compulsary, other values default to
%   {'name', initial,  -Inf, Inf,  NaN, Inf,  1,  0}
%
% OPTIONS mcmc run options
%    options.nsimu            number of simulations
%    options.qcov             proposal covariance
%    options.method           'dram','am','dr' or 'mh'
%    options.adaptint         interval for adaptation, if 'dram' or 'am' used
%                             DEFAULT adaptint = 100
%    options.drscale          scaling for proposal stages of dr
%                             DEFAULT 3 stages, drscale = [5 4 3]
%    options.updatesigma      update error variance. Sigma2 sampled with↙
updatesigma=1
%                             DEFAULT updatesigma=0
%    options.verbosity        level of information printed
%    options.waitbar          use graphical waitbar?
%    options.burnintime       burn in before adaptation starts
%
% Output:
%  RESULTS   structure that contains results and information about
%            the simulations
%  CHAIN, S2CHAIN, SSCHAIN
%           parameter, sigma2 and sum-of-squares chains

% Marko.Laine@helsinki.fi, 2003
% $Revision: 1.51 $  $Date: 2009/10/15 11:48:24 $

%% check input structs
goodopt={'nsimu','adaptint','ntry','method','printint',...
        'adaptend','lastadapt','burnintime','waitbar',...
        'debug','qcov','updatesigma','noadaptind','stats',...
↙
'drscale','adascale','savesize','maxmem','chainfile','s2chainfile',...
↙
'sschainfile','savedir','skip','label','RDR','verbosity','maxiter',...
    'priorupdatestart'};
goodmod={'sigma2','N','ssfun','modelfun','priorfun',...
     'priortype','priorupdatefun','priorpars','nbatch','S20','N0'};
[yn,bad]=checkoptions(options,goodopt);
if yn==0
  fprintf('bad options for mcmcrun:\n');
  for i=1:length(bad)
    fprintf('\t%s\n',bad{i});
  end
  fprintf('available options are:\n');
  for i=1:length(goodopt)
    fprintf('\t%s\n',goodopt{i});
  end
  error('please check options');
  return;
end
[yn,bad]=checkoptions(model,goodmod);
if yn==0
  fprintf('bad model options for mcmcrun:\n');
  for i=1:length(bad)
    fprintf('\t%s\n',bad{i});
  end
  fprintf('available options are:\n');
  for i=1:length(goodmod)
    fprintf('\t%s\n',goodmod{i});
  end
  error('please check model options');
  return;
end

%% set parameter defaults
%%% mcmc options
% some predefined methods
method = getpar(options,'method','dram');
switch lower(method)
 case 'mh'
  nsimu    = getpar(options,'nsimu',10000);  % length of the chain to↙
simulate
  adaptint = 0;
  Ntry     = 1;
 case 'am'
  nsimu    = getpar(options,'nsimu',10000);
  adaptint = getpar(options,'adaptint',100); % update interval for adaptation
  Ntry     = 1;
 case 'dr'
  nsimu    = getpar(options,'nsimu',10000);
  adaptint = 0;
  Ntry     = getpar(options,'ntry',2);        % DR tries (1 = no extra try)
 case 'dram'
  nsimu    = getpar(options,'nsimu',10000);
  adaptint = getpar(options,'adaptint',100);
  Ntry     = getpar(options,'ntry',2);
 otherwise
  error(sprintf('unknown mcmc method: %s',method));
end
printint    = getpar(options,'printint',NaN); % print interval
lastadapt   = getpar(options,'lastadapt',0);  % last adapt
lastadapt   = getpar(options,'adaptend',lastadapt);%  the same
burnintime  = getpar(options,'burnintime',0);
```

```
wbarupd     = getpar(options,'waitbar',1);    % use graphical waitbar
verbosity   = getpar(options,'verbosity',1);  % amout of info to print
shdebug     = getpar(options,'debug',0);      % show some debug information
qcov        = getpar(options,'qcov',[]);      % proposal covariance
updatesigma = getpar(options,'updatesigma',0);
noadaptind  = getpar(options,'noadaptind',[]); % do not adapt these indeses
dostats     = getpar(options,'stats',0);       % convergence statistics
% DR options
dodram   = getpar(options,'dram',0); % DR (not used, use ntry instead)
%DR_scale = getpar(options,'drscale',[60 30 15]);
DR_scale = getpar(options,'drscale',[5 4 3]);
adascale = getpar(options,'adascale',[]); % qcov_scale
if Ntry > 1, dodram=1; end

% save options
savesize   = getpar(options,'savesize',0); % rows of the chain in memory
if savesize <= 0 || savesize > nsimu
  savesize = nsimu;
end
maxmem      = getpar(options,'maxmem',0); % memory available in mega bytes
% temporary files if dumping to file
savedir     = getpar(options,'savedir',tempdir);
fnum = fix(rand*100000); % random number for the default filename
chainfile   = getpar(options,'chainfile',sprintf('chain_%05d.mat',fnum));
s2chainfile = getpar(options,'s2chainfile',sprintf('s2chain_%05d.mat',fnum));
sschainfile = getpar(options,'sschainfile',sprintf('sschain_%05d.mat',fnum));
skip        = getpar(options,'skip',1);
if ~isempty(savedir)
  chainfile   = [savedir,chainfile];
  s2chainfile = [savedir,s2chainfile];
  sschainfile = [savedir,sschainfile];
end
label = getpar(options,'label',sprintf('MCMC run at %s',date));

% Model options
sigma2  = getpar(model,'sigma2',[]);     % initial value for the error↙
variance
if isobject(data)
    N       = getpar(model,'N',getN(data.Data));  % no of obs
%elseif isstruct(data)
%    N       = getpar(model,'N',getN(data.Data));  % no of obs
else
    N       = getpar(model,'N',getN(data));  % no of obs
end
ssfun   = getpar(model,'ssfun',[]);      % sum of squares function
modelfun= getpar(model,'modelfun',[]);   % model function
priorfun= getpar(model,'priorfun',[]);   % prior function
priortype= getpar(model,'priortype',1);  % prior type, 1 = Gaussian
priorupdatefun = getpar(model,'priorupdatefun',[]); % prior parameter update
priorpars = getpar(model,'priorpars',[]); % prior parameter for↙
```

```
priorupdatefun
priorupdatestart = getpar(options,'priorupdatestart',burnintime);
%ssstyle = getpar(model,'ssstyle',1);
ssstyle = 1;
% error variance prior
S20      = getpar(model,'S20',NaN);
N0       = getpar(model,'N0',[]);

if isobject(data)
    nbatch   = getpar(model,'nbatch',getnbatch(data.Data));
%elseif isstruct(data)
%    nbatch   = getpar(model,'nbatch',getnbatch(data.Data));
else
    nbatch   = getpar(model,'nbatch',getnbatch(data)); % number of batches
end

if isempty(N)
  error('could not determine number of data points, please specify model.N');
end
if isempty(nbatch)
  message(verbosity,1,'Setting nbatch to 1\n');
  nbatch = 1;
end
% This is for backward compatibility
% if sigma2 given then default N0=1, else default N0=0
if isempty(N0)
  if isempty(sigma2)
    sigma2 = 1;
    N0 = 0;
  else
    N0 = 1;
  end
else
  % if N0 given, then also check updatesigma
  updatesigma = 1;
end

% some values from the previous run
if nargin > 4 && ~isempty(res)
  message(verbosity,0,'Using values from the previous run\n')
  params = res2par(res,params, 1 ); % 1 = do local parameters
  qcov   = res.qcov2;
end

% open and parse the parameter structure
[names,value,parind,local,upp,low,thetamu,thetasig,hyperpars] = ...
    openparstruct(params,nbatch);

if any(thetasig<=0)
  disp('some prior variances <=0, setting those to Inf')
  thetasig(thetasig<=0) = Inf;
end

% hyper prior parameters
hchain = []; % it is allocated after the first call inside the simuloop
if hyperpars.nhpar > 0
  fprintf('NOTE: n:o of parameters with hyper priors is %d\n',hyperpars.↙
nhpar);
  if isempty(priorpars), priorpars=hyperpars;end
  if isempty(priorupdatefun), priorupdatefun=@hyperpriorupdate;disp('  using↙
the default hyper update method');end
end

% default for sigma2 is S20 or 1
if isempty(sigma2)
  if not(isnan(S20))
    sigma2=S20;
  else
    sigma2=1;
  end
end
if isnan(S20)
  S20 = sigma2; % prior parameters for the error variance
end
if isnan(N0)
  N0 = 1;
end
if lastadapt<1
  lastadapt=nsimu;
end
if isnan(printint)
  printint = max(100,min(1000,adaptint));
end

if verbosity>0
  fprintf('Sampling these parameters:\nname   start [min,max] N(mu,s^2)\n');
  nprint = length(parind);
  if verbosity == 1
    nprint = min(nprint,40);
  end
  for i=1:nprint
    if ismember(i,noadaptind), st=' (*)'; else st='';end
    if isinf(thetasig(parind(i))), h2=''; else h2='^2';end
    fprintf('%s: %g [%g,%g] N(%g,%g%s)%s\n',...
            names{parind(i)},value(parind(i)),...
            low(parind(i)),upp(parind(i)),...
            thetamu(parind(i)),thetasig(parind(i)),h2,st);
  end
  if nprint < length(parind), fprintf('...\n'); end
end
```

```
par0 = value(parind);
npar = length(par0);

% check ssfun type
if isempty(ssfun)
  if isempty(modelfun)
    error('no ssfun or modelfun!')
  end
  ssstyle = 4;
  ni = 4;
else
  if isa(ssfun,'function_handle')
%    ni = nargin(func2str(ssfun)); % is this needed?
    ni = nargin(ssfun);
  elseif isa(ssfun,'inline') || exist(ssfun) == 2 % ssfun is an mfile
    ni = nargin(ssfun);
  else
    ni = 2;
  end
  if ni == 3
    ssstyle=2;
  elseif ni== 4
      ssstyle = 5;
  end
  if strcmp(char(model.ssfun),'ssobjectiveDRAM_gpr')
  ssstyle = 6; end
end

if isempty(qcov)
  qcov = thetasig.^2;
  ii = isinf(qcov)|isnan(qcov);
%  qcov(ii) = [abs(par0(ii))*0.05].^2; % default is 5% std
  qcov(ii) = [abs(value(ii))*0.05].^2; % default is 5% std
  qcov(qcov==0) = 1; % .. or one if we start from zero
  qcov = diag(qcov);
end

if isempty(adascale)||adascale<=0
  qcov_scale = 2.4 / sqrt(npar) ; % scale factor in R
else
  qcov_scale = adascale;
end
burnin_scale = 10; % scale in burn-in down/up
qcov_adjust  = 1e-5; % eps adjustment

[cm,cn]=size(qcov);
if min([cm cn]) == 1 % qcov contains variances!
  s = sqrt(qcov(parind));
  R = diag(s); % *qcov_scale; % do NOT scale the initial qcov
  qcovorig = diag(qcov); % save qcov
  qcov = diag(qcov(parind));
else %  qcov has covariance matrix in it
  qcovorig = qcov; % save qcov
  qcov = qcov(parind,parind);
  R    = chol(qcov); % *qcov_scale;
end
%R0 = R; % save R
global invR
global A_count
A_count = 0; % alphafun count
if dodram
  RDR = getpar(options,'RDR',[]); % RDR qiven in ooptions
  if ~isempty(RDR)
    for i=1:Ntry
      invR{i} = RDR{i}\eye(npar);
    end
    R = RDR{1};
  else
    % DR strategy: just scale R's down by DR_scale
    RDR{1} = R;
    invR{1} = R\eye(npar);
    for i=2:Ntry
      RDR{i}  = RDR{i-1}./DR_scale(min(i-1,length(DR_scale)));
      invR{i} = RDR{i}\eye(npar);
    end
  end
  iacce = zeros(1,Ntry);
end

starttime=clock;

oldpar=par0(:)';
ss = sseval(ssfun,ssstyle,oldpar,parind,value,local,data,modelfun);
ss1 = ss;
ss2 = ss;

ny = length(ss);
if length(S20)==1
  S20 = ones(1,ny)*S20;
end
if length(N)==1
  N = ones(1,ny)*N;
end
if length(N)==ny+1
  N = N(2:end); % remove first columns FIXME
end
if length(N0)==1
  N0 = ones(1,ny)*N0;
end
```

```
% default prior function calculates Gaussian sum of squares
if isempty(priorfun)
  priorfun = @(th,mu,sig) sum(((th-mu)./sig).^2);
end

oldprior = feval(priorfun,oldpar,thetamu(parind),thetasig(parind));

%memory calculations
memneeded = savesize*(npar+2*ny)*8*1e-6;
if (maxmem > 0) && (memneeded > maxmem)
  savesize = max(1000,floor(maxmem/(npar+2*ny)/8*1e6));
  message(verbosity,0,'savesize decreased to %d\n',savesize);
end
if (savesize < nsimu) || (nargout < 2)
  saveit = 1;
else
  saveit = 0;
end
% save parameters, error variance, and SS
chain   = zeros(savesize,npar);
if updatesigma
  s2chain = zeros(savesize,ny);
else
  s2chain = [];
end
sschain = zeros(savesize,ny);

%% save chain
if saveit == 1
  savebin(chainfile,[],'chain');
  savebin(sschainfile,[],'sschain');
  if updatesigma
    savebin(s2chainfile,[],'s2chain');
  end
end

chain(1,:)   = oldpar;
if updatesigma
  s2chain(1,:) = sigma2;
end
sschain(1,:) = ss;

rej=0; reju=0; ii=1; rejl = 0;
%% setup waitbar
if wbarupd; wbar('init'); end

% covariance update uses these to store previous values
covchain = []; meanchain = []; wsum = []; lasti = 0;
% no update for these indeses
```

```
noupd = logical(zeros(1,npar));
noupd(intersect(parind,noadaptind)) = 1;

chainind = 1; % where we are in chain
for isimu=2:nsimu % simulation loop
  ii = ii+1; % local adaptation index (?)
  chainind = chainind+1;

  % waitbar
  if wbarupd;
    status = wbar('',isimu,nsimu);
    if status == -1 % waitbar killed, cancel the run and keep
                    % the chain so far
      message(verbosity,1,'Cancelling...\n');
      chainind = chainind-1;
      nsimu = isimu;
      chain = chain(1:chainind,:);
      sschain = sschain(1:chainind,:);
      if updatesigma
        s2chain = s2chain(1:chainind,:);
      end
      if size(hchain,1)>1
        hchain = hchain(1:chainind,:);
      end
      break % break the nsimu loop
    end
  end
  message(verbosity,100,'i:%d/%d\n',isimu,nsimu);

  % sample new candidate from Gaussian proposal
  newpar=oldpar+randn(1,npar)*R;

  % reject points outside boundaries
  if any(newpar<low(parind)) || any(newpar>upp(parind))
    accept = 0;
    newprior = 0;
    tst      = 0;
    ss1      = Inf;
    ss2      = ss;
    outbound = 1;
  else
    outbound = 0;
    % prior SS for the new theta
    newprior = feval(priorfun,newpar,thetamu(parind),thetasig(parind));

    % calculate ss
    ss2 = ss;             % old ss
    ss1 = sseval(ssfun,ssstyle,newpar,parind,value,local,data,modelfun);

    tst = exp(-0.5*( sum((ss1-ss2)./sigma2) + newprior-oldprior) );
    if tst <= 0
      accept = 0;
    elseif tst >= 1
      accept = 1;
    elseif tst > rand(1,1)
      accept = 1;
    else
      accept = 0;
    end
    if shdebug && fix(isimu/shdebug) == isimu/shdebug
      fprintf('%d: pri: %g, tst: %g, ss: %g\n',isimu, newprior,tst, ss1);
    end
  end
  %%% DR -----------------------------------------------------
  if dodram == 1 && accept == 0 % & outbound == 0
    % we do a new try according to delayed rejection
    x.p   = oldpar;
    x.ss  = ss2;
    x.pri = oldprior;
    x.s2  = sigma2;

    y.p   = newpar;
    y.ss  = ss1;
    y.pri = newprior;
    y.s2  = sigma2;
    y.a   = tst;

    trypath = {x,y};
    itry    = 1;
    while accept == 0 & itry < Ntry
      itry = itry+1;
      z.p  = x.p + randn(1,npar)*RDR{itry};
      z.s2 = sigma2;
      if any(z.p<low(parind)) || any(z.p>upp(parind))
        z.a   = 0;
        z.pri = 0;
        z.ss  = Inf;
        trypath = {trypath{:},z};
        outbound = 1;
        continue
      end

      outbound = 0;
      z.ss = sseval(ssfun,ssstyle,z.p,parind,value,local,data,modelfun);
      z.pri = feval(priorfun,z.p,thetamu(parind),thetasig(parind));
      trypath = {trypath{:},z};
      alpha = alphafun(trypath{:});
      trypath{end}.a = alpha;
      if alpha >= 1 || rand(1,1) < alpha     %  accept
```

```
        accept   = 1;
        newpar   = z.p;
        ss1      = z.ss;
        newprior = z.pri;
        iacce(itry) = iacce(itry) + 1;
      end
      if shdebug && fix(isimu/shdebug) == isimu/shdebug
        fprintf('try %d: pri: %g, alpha: %g\n',itry, z.pri, alpha);
        fprintf(' p: %g\n',z.p);
      end
    end
  end % DR -------------------------------------------------------
  %%% save chain
  if accept
    %%% accept
    chain(chainind,:) = newpar;
    oldpar      = newpar;
    oldprior    = newprior;
    ss          = ss1;
  else
    %%%% reject
    chain(chainind,:) = oldpar;
    rej         = rej + 1;
    reju        = reju + 1;
    if outbound
      rejl      = rejl + 1;
    end
  end
  %%% Possibly update the prior parameters (for testing hiearchical hyper↙
priors)
  %%% [mu,sig]=priorupdatefun(theta, mu, sig, priorpars)
  if not(isempty(priorupdatefun))
    if isimu==2 || isimu>=priorupdatestart
      [muout,sigout,hrowout] = ...
      feval(priorupdatefun,oldpar,thetamu(parind),thetasig(parind),↙
priorpars);
      if isimu==2 % set up hchain
        hchain = zeros(nsimu,length(hrowout));
    if isfield(priorpars,'mu0') && isfield(priorpars,'sig20') && ...
          length([priorpars.mu0,priorpars.sig20]) == length(hrowout)
      hchain(1,1:2:end) = priorpars.mu0;
      hchain(1,2:2:end) = sqrt(priorpars.sig20);
      hrow = hchain(1,:);
    end
      end
      if isimu>=priorupdatestart; % update mu and theta
    thetamu(parind)  = muout;
    thetasig(parind) = sigout;
    hrow = hrowout;
    % need to update the prior ss value
    oldprior = feval(priorfun,oldpar,thetamu(parind),thetasig(parind));
      end
    end
    hchain(isimu,:) = hrow;
    %% fix this:
    %% (do?) we need "sum of squares" of the hyper parameters for the↙
observation
    %% noise sigma2 update
%    sig2s = hrow(2:2:end).^2; % now assumes that we are using the default↙
function
%    sssig = sum(sig2s);
%    sign = length(sig2s)*nbatch;
  else
%    sssig = 0;
%    sign = 0;
  end
  %%%
  %%% update sigma2
  if updatesigma
    for j=1:ny
     sigma2(j) = 1./gammar(1,1,(N0(j)+N(j))/2,2./(N0(j).*S20(j)+ss(j)));
 %     nn = N0(j)+N(j)+sign;
 %     sigma2(j) = invchir(1,1, nn , (N0(j).*S20(j)+ss(j) + sssig)./nn);
 %     sigma2(j) =  1./gammar(1,1,(N0(j)+N(j)+sign)/2,2./(N0(j).*S20(j)+ss(j)↙
+sssig ));
    end
    s2chain(chainind,:) = sigma2;
  end
  %%%
  sschain(chainind,:) = ss;
  %
  if printint && fix(isimu/printint) == isimu/printint
    message(verbosity,2,'i:%g (%3.2f,%3.2f,%3.2f)\n', ...
            isimu,rej/isimu*100,reju/ii*100,rejl/isimu*100);
  end

  %% adaptation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  if adaptint>0 && isimu<=lastadapt && fix(isimu/adaptint) == isimu/adaptint
    if isimu < burnintime
      % During burnin no adaptation, just scaling up/down
      if reju/ii > 0.95
        message(verbosity,2,' (burnin/down) %3.2f',reju/ii*100);
        R = R./burnin_scale;
      elseif reju/ii < 0.05
        message(verbosity,2,' (burnin/up) %3.2f',reju/ii*100)
        R = R.*burnin_scale;
      end
    else
      message(verbosity,2,'i:%g adapting (%3.2f,%3.2f,%3.2f)', ...
              isimu,rej/isimu*100,reju/ii*100,rejl/isimu*100);
      [covchain,meanchain,wsum] = covupd(chain((lasti+1):chainind,1:npar),1,↙
...
                                         covchain,meanchain,wsum);
      lasti = chainind;
      %%%
      upcov          = covchain;
      upcov(noupd,:) = qcov(noupd,:);
      upcov(:,noupd) = qcov(:,noupd);
      %%%
      [Ra,p] = chol(upcov);
      if p % singular cmat
        % try to blow it
        [Ra,p] = chol(upcov + eye(npar)*qcov_adjust);
        if p % stil singular
          message(verbosity,0,' (cmat singular, no adapt) %3.2f',↙
reju/ii*100);
        else
          message(verbosity,2,' [adjusted cmat]');
          % scale R
          R = Ra * qcov_scale;
        end
      else
        R = Ra * qcov_scale;
      end
      lasti = isimu;
      if dodram  %%%% delayed rejection
        RDR{1} = R;
        invR{1} = RDR{1}\eye(npar);
        for k=2:Ntry
          RDR{k}  = RDR{k-1}./DR_scale(min(k-1,length(DR_scale)));
          invR{k} = invR{k-1}.*DR_scale(min(k-1,length(DR_scale)));
        end
      end
    end
    message(verbosity,2,'\n');
    reju = 0; ii = 0;
  end
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %%% save chain
  if chainind == savesize && saveit == 1
    message(verbosity,2,'saving chains\n');
    addbin(chainfile,chain');
    addbin(sschainfile,sschain');
    if updatesigma
      addbin(s2chainfile,s2chain');
    end
    chainind = 0;
    % update covariance
    [covchain,meanchain,wsum] = covupd(chain((lasti+1):chainind,1:npar),1,↙
...
```

```
                                       covchain,meanchain,wsum);
    lasti = 0;
  end

end % nsimu

% save the rest
if chainind>0 && saveit == 1
  addbin(chainfile,chain(1:chainind,:)');
  addbin(sschainfile,sschain(1:chainind,:)');
  if updatesigma
    addbin(s2chainfile,s2chain(1:chainind,:)');
  end
  % update covariance
  [covchain,meanchain,wsum] = covupd(chain((lasti+1):chainind,1:npar),1, ...
                                     covchain,meanchain,wsum);
end

if wbarupd; wbar('close'); end

value(parind) = oldpar; % update the initial value to the final value

%% build the results structure
if nargout>0
  results.class = 'MCMC';
  results.label = label;
  results.method = method;
  results.rejected   = rej/nsimu;
  results.ulrejected = rejl/nsimu;
  results.R      = R;
  results.qcov   = R'*R; % with scale %  ./ qcov_scale.^2 ;
  qcovorig(parind,parind) = results.qcov;
  results.qcov2  = qcovorig; % original size
  results.cov    = covchain;
  results.mean   = meanchain;
  results.names  = names(parind);
  results.limits = [low(parind)',upp(parind)'];
  results.prior  = [thetamu(parind)',thetasig(parind)'];
  results.theta  = value; % last values
  results.parind = parind;
  results.local  = local;
  results.nbatch = nbatch;
  results.N      = N;
  if updatesigma
    results.sigma2 = NaN;
    results.S20    = S20;
    results.N0     = N0;
  else
    results.sigma2 = sigma2;
    results.S20    = NaN;
    results.N0     = NaN;
  end
  results.modelfun = modelfun;
  results.ssfun    = ssfun;
  results.priorfun = priorfun;
  results.priortype= priortype;
  results.priorpars= priorpars;
  results.nsimu    = nsimu;
  results.adaptint = adaptint;
  results.adaptend = lastadapt;
  results.adascale = adascale;
  results.skip     = skip;
  results.simutime = etime(clock,starttime);
  results.ntry     = Ntry;
  if dodram
    results.ntry  = Ntry;
    results.drscale = DR_scale; % .^2;
    iacce(1) = nsimu-rej-sum(iacce(2:end));
    results.iacce = iacce;
    results.alpha_count = A_count;
    results.RDR = RDR;
  end
end

% check if we need to read the generated chain from binary dump files
if saveit == 1 && savesize < nsimu
  if nargout > 1
    chain = readbin(chainfile,1,skip);
  end
  if nargout > 2 && updatesigma
    s2chain = readbin(s2chainfile,1,skip);
  end
  if nargout > 3
    sschain = readbin(sschainfile,1,skip);
  end
elseif skip>1&&skip<=nsimu
  chain = chain(1:skip:end,:);
  if updatesigma
    s2chain = s2chain(1:skip:end,:);
  end
  sschain = sschain(1:skip:end,:);
end
% calculate some extra statistics (we need the whole chain to do this)
if dostats && (saveit == 1 || savesize >= nsimu)
  results.tau    = iact(chain);
  results.geweke = geweke(chain);
  results.rldiag = rldiag(chain);
  %% calculate DIC = 2*mean(ss)-ss(mean(chain))
  ss = sseval(ssfun,ssstyle,meanchain,parind,value,local,data,modelfun);
  D = mean(sschain);
  results.dic  = 2*D-ss; % Deviance Information Criterion
  results.pdic = D-ss;   % Effective number of parameters
end
%% end of main function
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function ss = sseval(ssfun,ssstyle,theta,parind,value,local,data,modelfun)
% evaluate the "sum-of-squares" function
value(parind) = theta;
if ssstyle == 1
  ss = feval(ssfun,value(:)',data);
elseif ssstyle == 4
  ss = mcmcssfun(value(:)',data,local,modelfun);
elseif ssstyle == 5
  ss = feval(ssfun,value(:)',data,local,parind);
elseif ssstyle == 6
  ss = feval(ssfun,theta,data);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function y=alphafun(varargin)
% alphafun(x,y1,y2,y3,...)
% recursive acceptance function for delayed rejection
% x.p, y1.p, ... contain the parameter value
% x.ss, y1.ss, ... the sum of squares
% x.a, y1.a, ... past alpha probabilities

% ML 2003

global A_count
A_count = A_count+1;

stage = nargin - 1; % The stage we're in, elements in varargin - 1
% recursively compute past alphas
a1 = 1; a2=1;
for k=1:stage-1
%  a1 = a1*(1-varargin{k+1}.a); % already have these alphas
% Thanks to E. Prudencio for pointing out an error here
  a1 = a1*(1-alphafun(varargin{1:(k+1)}));
  a2 = a2*(1-alphafun(varargin{(stage+1):-1:(stage+1-k)}));
  if  a2==0  % we will came back with prob 1
    y = 0;
    return
  end
end
y = lfun(varargin{1},varargin{end});
for k=1:stage
  y = y + qfun(k,varargin{:});
end
y = min(1, exp(y).*a2./a1);
%************************************************************%
function z=qfun(iq,varargin)
```

```
% Gaussian n:th stage log proposal ratio
% log of q_i(y_n,..,y_n-j) / q_i(x,y_1,...,y_j)

global invR

stage = nargin-1-1;
if stage == iq
  z = 0;                                % we are symmetric
else
  iR = invR{iq};                        % proposal^(-1/2)
  y1 = varargin{1}.p;           % y1
  y2 = varargin{iq+1}.p;        % y_i
  y3 = varargin{stage+1}.p;     % y_n
  y4 = varargin{stage-iq+1}.p;  % y_(n-i)
  z = -0.5*(norm((y4-y3)*iR)^2-norm((y2-y1)*iR)^2);
end
%************************************************************%
function z=lfun(x,y)
% log posterior ratio,  log( pi(y)/pi(x) * p(y)/p(x) )
z = -0.5*( sum((y.ss./y.s2-x.ss./x.s2)) + y.pri - x.pri );
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function message(verbosity,level,fmt,varargin)
if verbosity>=level
  fprintf(fmt,varargin{:})
end
if level<=2&&~strcmp(fmt,'\n')
  wbar('message',sprintf(fmt,varargin{:}));
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function status=wbar(task,i,nsimu)
%%%% waitbar update
persistent hdl t0 tl hmsg

status = 1;

switch lower(task)
 case 'init'
  hdl=waitbar(0,'Generating chain...','CreateCancelBtn','delete(gcbf)');
  set(hdl,'Name','MCMC status');
  t0=clock;
  tl=t0;
  hmsg=get(findobj(hdl,'Type','Axes'),'xlabel');
  set(hmsg,'HorizontalAlignment','left');
  set(hmsg,'Position',[0,-1]);
 case 'close'
  if ishandle(hdl);delete(hdl);end
 case 'message'
  if ishandle(hdl)
    txt = i;
    set(hmsg,'String',txt);
    drawnow
  end
 otherwise
  if ~ishandle(hdl) % cancel pressed
    status = -1;
    return
  end
  if (i/50==fix(i/50))
    % too slow
%  if etime(clock,tl) >= 1 | i < 10 % update every 1 secs
    hh=i/nsimu;
    %    htitle=get(findobj(hdl,'Type','Axes'),'title');
    secs = etime(clock,t0)*(1-hh)/hh;
    mins = floor(secs/60);
    secs = ceil(secs - 60*mins);
    hrs  = floor(mins/60);
    mins = mins - hrs*60;
    %   if wbarupd
    waitbar(hh,hdl, ...
            sprintf('Generating chain, eta: %g:%02g:%02g', ...
                    hrs,mins,secs));
%    set(htitle,'String', ...
%               sprintf('Generating chain, eta: %g:%02g:%02g', ...
%                       hrs,mins,secs));
    drawnow
    tl = clock; % last time updated
  end
end
%%%%% EOF %%%%
```

```
function [out]=fcnSensitivityRun(Objfun, BCcoeff, setts, PNM, likemisfit)
% Performes sensitivity correlation check by pairs for any function

% Objfun- objective function with one scalar return
% BCcoeff- base case parameter values
% setts -settings e.g. range,step
% param - additional parameters for the Objfun function
% returns X,Y,Z cells for each parameter pair.

% ver. 1.0
% by Levente L. SIMON - ETH Zuerich
% email: l.simon@chem.ethz.ch

% Modified by
% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

Rangesize = setts.Range;
Step = setts.step;

% Location of the parameters to be optimized
optidx=targetvector(PNM);
opt_param=PNM.Parameters(optidx(1,:));

% Open the PNM data structure of the parameters
[names,value,parind,local,upp,low] = ...
    openparstruct(opt_param,length(PNM.Data));
names{4} = 'sigma';
sigma_L = 2.79; sigma_U = 2.80; % std measur. errors
low = [low    sigma_L];
upp = [upp    sigma_U];

% Set step
Srange = (-1) * Rangesize :Step: Rangesize;
for i =1: size(BCcoeff,2)
    % Calculate the interval for each coefficient
    Coeffrange(i,:)  = BCcoeff(i) + (Srange./100 .* BCcoeff(i)) ;
    if setts.posparam==true
        % Upper and lower bounds
        Coeffrange(i,Coeffrange(i,:)<=low(i))=low(i);
        Coeffrange(i,Coeffrange(i,:)>=upp(i))=upp(i);
    end
end

% Calculation of combinations based on the parameter number
CoeffComb2 = combnk(1:size(BCcoeff,2), 2);
% Sorts matrix from back to front , v 1.1
CoeffComb = CoeffComb2(end:-1:1,:);
```

```
% Plot settings
MaxSubPlot = ceil(size(CoeffComb,1) / 3);
for n=1:size(CoeffComb,1)
    % Get interval for each data pair
    X = Coeffrange (CoeffComb(n,1), :);
    Y = Coeffrange (CoeffComb(n,2), :);
    clear Z
    for i=1:size(Y,2) % have allways same size, simetric
        for j=1:size(X,2)
            % update vector of BC coefficients
            Scoeff = BCcoeff; Scoeff(CoeffComb(n,1)) = X(j);
            Scoeff(CoeffComb(n,2)) = Y(i);
            Z(j,i)=feval(Objfun, Scoeff, PNM, likemisfit);
        end
    end
    % Normalize the Z
    Zz = fcnNormalize0_1(Z);
    Xcell{n} = X; Ycell{n} = Y; Zcell{n} = Zz;
    XlabelCell{n} = names{CoeffComb(n,1)};
    YlabelCell{n} = names{CoeffComb(n,2)};
    disp(strcat('Z matrix calculated for pair:',num2str(n)));
end

% Function output structure
out.Xcell = Xcell;
out.Ycell =Ycell;
out.Zcell=Zcell;
out.XlabelCell=XlabelCell;
out.YlabelCell=YlabelCell;
out.MaxSubPlot=MaxSubPlot;
out.setts=setts;

function y= fcnNormalize0_1(a)
% normalizes matrix between 100 and 0; input, output: matrix
% higheest value = 0; lowest = 100;
% use y= fcnNormalize0_1(-a) for high=100; low=0;
b=[];
for i=1:size(a)
    b = [b a(i,:)]; % converts matrix to a vector
end
n=(b-(min(b)))/max(b-min(b)); % normalizes between 0 and 1
n = abs(n-1);                 % changes the normaliation between 1 and 0
n= n.*100;                     % range = 0 :100
y = reshape ( n,size(a,2),size(a,2)); % converts vector to same size matrix
```

```
>>
```

```
function y=acf(x,lagmax)
%ACF Autocorrelation function
% ACF(X,maxlag)
% default maxlag is floor(10*log10(length(x)))

% There is also acf.mex version which is much faster
% it is used if Matlab finds it

% Marko Laine <Marko.Laine@Helsinki.FI>
% $Revision: 1.4 $  $Date: 2007/08/09 13:42:26 $

x = x(:)'-mean(x);
n = length(x);
if nargin<2
  lagmax = floor(10*log10(n));
  lagmax = min(lagmax, n-1);
end
y = filter(x(n:-1:1),1,x);
%y  = conv(flipud(x),x);
y = y(n:-1:1)/n;
y = y/y(1);
y = y(1:lagmax+1);
```

```
function [stats,stats2]=chainstats(chain,results,fid)
%CHAINSTATS some statistics from the MCMC chain
% chainstats(chain,results)
%    chain    nsimu*npar MCMC chain
%    results  output from mcmcrun function

% $Revision: 1.4 $  $Date: 2009/08/13 15:47:35 $

% Modified by
% Zivko Juznic-Zonta
% Copyrigth,10-Jul-2012
% Mollet del Valles, BCN, Spain
% UPC/GIRO-CT

if nargin<3, fid=1; end % fid=1, standard output

if nargin>1
  if isstruct(results) % results struct from mcmcrun
    names=results.names;
  else
    names = results; % char array of names
  end
end
d=tinv(1-.05/2,size(chain,1)-size(chain,2));
mcerr = bmstd(chain)./sqrt(size(chain,1));

[z,p]  = geweke(chain);
tau    = iact(chain);
stats  = [mean(chain)',std(chain)',std(chain)'.*d,mcerr',tau', p'];

[m,n] = size(stats);

fprintf(fid,'MCMC statistics, nsimu = %g\n\n', size(chain,1));

if nargin>1
  fprintf(fid,'% 10s ','');
end
fprintf(fid,'% 10s  % 10s  % 10s %10s  % 10s  %↙
10s\n','mean','std','CI95%','MC_err','tau','geweke');
if nargin>1
  fprintf(fid,'-----------');
end
fprintf↙
(fid,'-----------------------------------------------------------------------↙
\n');
for i = 1:m
  if nargin>1
    fprintf(fid,'% 10s ',names{i});
  end
  fprintf(fid,'%10.5g  %10.5g %10.5g  %10.5g  %10.5g  %10.5g\n',stats(i,:));
```

```
end
if nargin>1
  fprintf(fid,'-----------');
end
fprintf↙
(fid,'-----------------------------------------------------------------------↙
\n');
fprintf(fid,'\n');

if nargout>1

%% more statistics (NOT printed now)

for i=1:m
    [f,xi] = ksdensity(chain(:,i));
    %[ans,f,xi] = kde(chain(:,i));
    modei(i,1) = xi(f==max(f));
end
l      = quantile(chain,[0.025,0.5,0.975]);
stats2 = [modei, min(chain)', l',max(chain)',abs(l(1,:)-l(3,:))'];

if nargin>1
  fprintf(fid,'% 10s ','');
end
fprintf(fid,'% 10s % 10s  % 10s  % 10s  % 10s  % 10s %↙
10s\n','mode','min','Qinf','med','Qsup','max','Qrange');
if nargin>1
  fprintf(fid,'-----------');
end
fprintf↙
(fid,'-----------------------------------------------------------------------↙
----------\n');
for i = 1:m
  if nargin>1
    fprintf(fid,'% 10s ',names{i});
  end
  fprintf(fid,'%10.5g %10.5g  %10.5g  %10.5g  %10.5g  %10.5g %10.5g\n',stats2↙
(i,:));
end
if nargin>1
  fprintf(fid,'-----------');
end
fprintf↙
(fid,'-----------------------------------------------------------------------↙
----------\n');
fprintf(fid,'\n');

end
```

```
function h=error_ellipse(varargin)
% ERROR_ELLIPSE - plot an error ellipse, or ellipsoid, defining confidence↙
region
%    ERROR_ELLIPSE(C22) - Given a 2x2 covariance matrix, plot the
%    associated error ellipse, at the origin. It returns a graphics handle
%    of the ellipse that was drawn.
%
%    ERROR_ELLIPSE(C33) - Given a 3x3 covariance matrix, plot the
%    associated error ellipsoid, at the origin, as well as its projections
%    onto the three axes. Returns a vector of 4 graphics handles, for the
%    three ellipses (in the X-Y, Y-Z, and Z-X planes, respectively) and for
%    the ellipsoid.
%
%    ERROR_ELLIPSE(C,MU) - Plot the ellipse, or ellipsoid, centered at MU,
%    a vector whose length should match that of C (which is 2x2 or 3x3).
%
%    ERROR_ELLIPSE(...,'Property1',Value1,'Name2',Value2,...) sets the
%    values of specified properties, including:
%      'C' - Alternate method of specifying the covariance matrix
%      'mu' - Alternate method of specifying the ellipse (-oid) center
%      'conf' - A value betwen 0 and 1 specifying the confidence interval.
%        the default is 0.5 which is the 50% error ellipse.
%      'scale' - Allow the plot the be scaled to difference units.
%      'style' - A plotting style used to format ellipses.
%      'clip' - specifies a clipping radius. Portions of the ellipse, -oid,
%        outside the radius will not be shown.
%
%    NOTES: C must be positive definite for this function to work properly.

default_properties = struct(...
  'C', [], ... % The covaraince matrix (required)
  'mu', [], ... % Center of ellipse (optional)
  'conf', 0.5, ... % Percent confidence/100
  'scale', 1, ... % Scale factor, e.g. 1e-3 to plot m as km
  'style', '', ...  % Plot style
  'clip', inf); % Clipping radius

if length(varargin) >= 1 & isnumeric(varargin{1})
  default_properties.C = varargin{1};
  varargin(1) = [];
end

if length(varargin) >= 1 & isnumeric(varargin{1})
  default_properties.mu = varargin{1};
  varargin(1) = [];
end

if length(varargin) >= 1 & isnumeric(varargin{1})
  default_properties.conf = varargin{1};
  varargin(1) = [];
```

```
end

if length(varargin) >= 1 & isnumeric(varargin{1})
  default_properties.scale = varargin{1};
  varargin(1) = [];
end

if length(varargin) >= 1 & ~ischar(varargin{1})
  error('Invalid parameter/value pair arguments.')
end

prop = getopt(default_properties, varargin{:});
C = prop.C;

if isempty(prop.mu)
  mu = zeros(length(C),1);
else
  mu = prop.mu;
end

conf = prop.conf;
scale = prop.scale;
style = prop.style;

if conf <= 0 | conf >= 1
  error('conf parameter must be in range 0 to 1, exclusive')
end

[r,c] = size(C);
if r ~= c | (r ~= 2 & r ~= 3)
  error(['Don''t know what to do with ',num2str(r),'x',num2str(c),' matrix'])
end

x0=mu(1);
y0=mu(2);

% Compute quantile for the desired percentile
k = sqrt(qchisq(conf,r)); % r is the number of dimensions (degrees of↙
freedom)

hold_state = get(gca,'nextplot');

if r==3 & c==3
  z0=mu(3);

  % Make the matrix has positive eigenvalues - else it's not a valid↙
covariance matrix!
  if any(eig(C) <=0)
    error('The covariance matrix must be positive definite (it has non-↙
positive eigenvalues)')
```

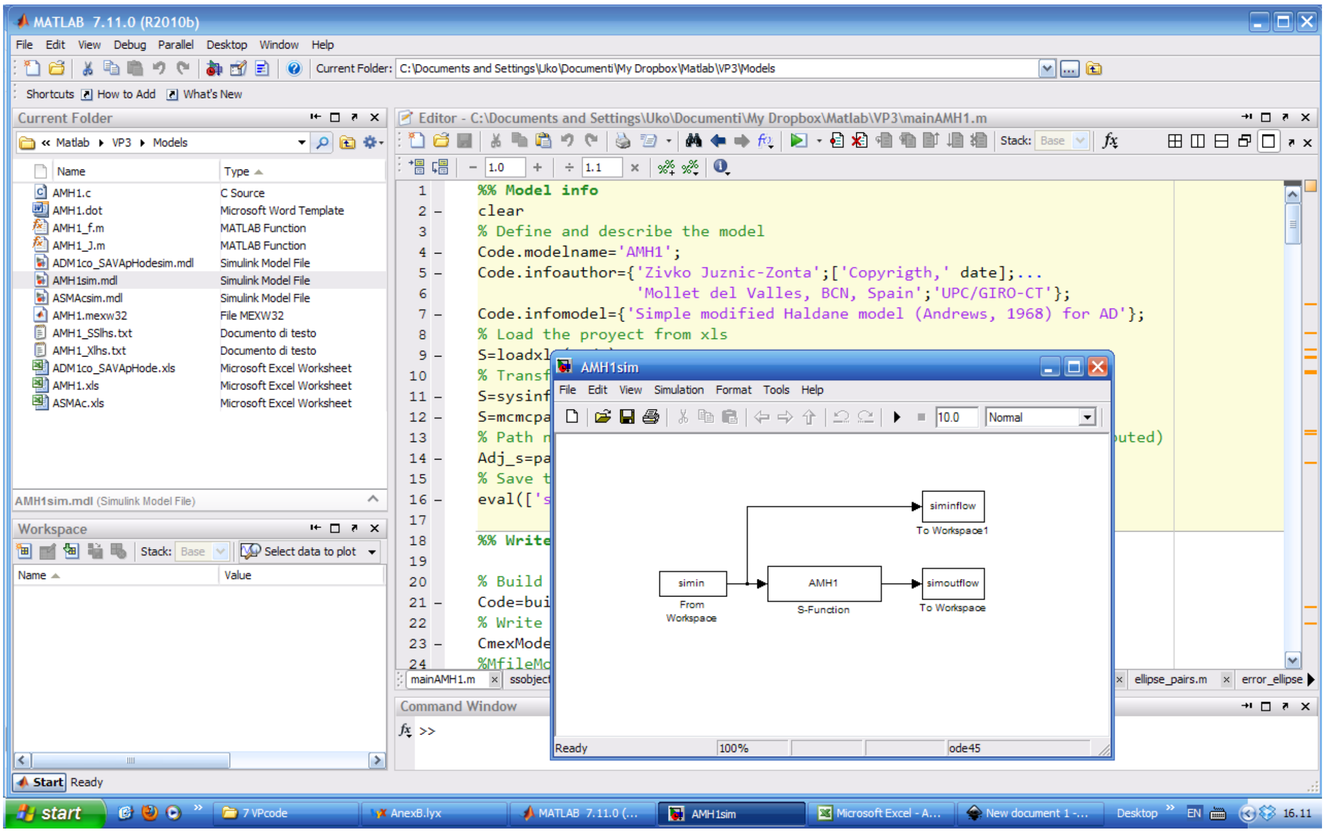
```
 end

 % C is 3x3; extract the 2x2 matricies, and plot the associated error
 % ellipses. They are drawn in space, around the ellipsoid; it may be
 % preferable to draw them on the axes.
 Cxy = C(1:2,1:2);
 Cyz = C(2:3,2:3);
 Czx = C([3 1],[3 1]);

 [x,y,z] = getpoints(Cxy,prop.clip);
 h1=plot3(x0+k*x,y0+k*y,z0+k*z,prop.style);hold on
 [y,z,x] = getpoints(Cyz,prop.clip);
 h2=plot3(x0+k*x,y0+k*y,z0+k*z,prop.style);hold on
 [z,x,y] = getpoints(Czx,prop.clip);
 h3=plot3(x0+k*x,y0+k*y,z0+k*z,prop.style);hold on


 [eigvec,eigval] = eig(C);

 [X,Y,Z] = ellipsoid(0,0,0,1,1,1);
 XYZ = [X(:),Y(:),Z(:)]*sqrt(eigval)*eigvec';

 X(:) = scale*(k*XYZ(:,1)+x0);
 Y(:) = scale*(k*XYZ(:,2)+y0);
 Z(:) = scale*(k*XYZ(:,3)+z0);
 h4=surf(X,Y,Z);
 colormap gray
 alpha(0.3)
 camlight
 if nargout
  h=[h1 h2 h3 h4];
 end
elseif r==2 & c==2
 % Make the matrix has positive eigenvalues - else it's not a valid↙
covariance matrix!
 if any(eig(C) <=0)
  error('The covariance matrix must be positive definite (it has non-↙
positive eigenvalues)')
 end

 [x,y,z] = getpoints(C,prop.clip);
 h1=plot(scale*(x0+k*x),scale*(y0+k*y),prop.style);
 set(h1,'zdata',z+1)
 if nargout
  h=h1;
 end
else
 error('C (covaraince matrix) must be specified as a 2x2 or 3x3 matrix)')
end
%axis equal
```

```
set(gca,'nextplot',hold_state);

%-------------------------------------------------------------
% getpoints - Generate x and y points that define an ellipse, given a 2x2
%   covariance matrix, C. z, if requested, is all zeros with same shape as
%   x and y.
function [x,y,z] = getpoints(C,clipping_radius)

n=100; % Number of points around ellipse
p=0:pi/n:2*pi; % angles around a circle

[eigvec,eigval] = eig(C); % Compute eigen-stuff
xy = [cos(p'),sin(p')] * sqrt(eigval) * eigvec'; % Transformation
x = xy(:,1);
y = xy(:,2);
z = zeros(size(x));

% Clip data to a bounding radius
if nargin >= 2
  r = sqrt(sum(xy.^2,2)); % Euclidian distance (distance from center)
  x(r > clipping_radius) = nan;
  y(r > clipping_radius) = nan;
  z(r > clipping_radius) = nan;
end

%-------------------------------------------------------------
function x=qchisq(P,n)
% QCHISQ(P,N) - quantile of the chi-square distribution.
if nargin<2
  n=1;
end

s0 = P==0;
s1 = P==1;
s = P>0 & P<1;
x = 0.5*ones(size(P));
x(s0) = -inf;
x(s1) = inf;
x(~(s0|s1|s))=nan;

for ii=1:14
  dx = -(pchisq(x(s),n)-P(s))./dchisq(x(s),n);
  x(s) = x(s)+dx;
  if all(abs(dx) < 1e-6)
    break;
  end
end

%-------------------------------------------------------------
```

```
function F=pchisq(x,n)
% PCHISQ(X,N) - Probability function of the chi-square distribution.
if nargin<2
  n=1;
end
F=zeros(size(x));

if rem(n,2) == 0
  s = x>0;
  k = 0;
  for jj = 0:n/2-1;
    k = k + (x(s)/2).^jj/factorial(jj);
  end
  F(s) = 1-exp(-x(s)/2).*k;
else
  for ii=1:numel(x)
    if x(ii) > 0
      F(ii) = quadl(@dchisq,0,x(ii),1e-6,0,n);
    else
      F(ii) = 0;
    end
  end
end

%---------------------------------------------------------------
function f=dchisq(x,n)
% DCHISQ(X,N) - Density function of the chi-square distribution.
if nargin<2
  n=1;
end
f=zeros(size(x));
s = x>=0;
f(s) = x(s).^(n/2-1).*exp(-x(s)/2)./(2^(n/2)*gamma(n/2));

%---------------------------------------------------------------
function properties = getopt(properties,varargin)
%GETOPT - Process paired optional arguments as 'prop1',val1,'prop2',val2,...
%
%   getopt(properties,varargin) returns a modified properties structure,
%   given an initial properties structure, and a list of paired arguments.
%   Each argumnet pair should be of the form property_name,val where
%   property_name is the name of one of the field in properties, and val is
%   the value to be assigned to that structure field.
%
%   No validation of the values is performed.
%
% EXAMPLE:
%   properties = struct('zoom',1.0,'aspect',1.0,'gamma',1.0,'file',[],'bg',↙
[]);
%   properties = getopt(properties,'aspect',0.76,'file','mydata.dat')
% would return:
%   properties =
%         zoom: 1
%       aspect: 0.7600
%        gamma: 1
%         file: 'mydata.dat'
%           bg: []
%
% Typical usage in a function:
%   properties = getopt(properties,varargin{:})

% Process the properties (optional input arguments)
prop_names = fieldnames(properties);
TargetField = [];
for ii=1:length(varargin)
  arg = varargin{ii};
  if isempty(TargetField)
    if ~ischar(arg)
      error('Propery names must be character strings');
    end
    f = find(strcmp(prop_names, arg));
    if length(f) == 0
      error('%s ',['invalid property ''',arg,'''; must be one of:'],↙
prop_names{:});
    end
    TargetField = arg;
  else
    % properties.(TargetField) = arg; % Ver 6.5 and later only
    properties = setfield(properties, TargetField, arg); % Ver 6.1 friendly
    TargetField = '';
  end
end
if ~isempty(TargetField)
  error('Property names and values must be specified in pairs.');
end
```

```
function hc=panellims(x,y,smo,rho,dens,ccolor)
%PANELLIMS 2d density with probability limits added to pairs plot. See PAIRS.
% panellims(x,y,smo,rho,dens,ccolor)
% smo - smoothing factor
% rho - correlation coef for the kernel
% dens - if 1, add marginal densities
% ccolor - contour color

% Marko Laine <Marko.Laine@Helsinki.FI>
% $Revision: 1.8 $  $Date: 2008/01/23 08:11:03 $

if nargin < 3
   smo = [1 1];
end
if length(smo)<2
   smo = [smo,smo];
end
if nargin<4
  rho = [];
end
if nargin<5
  dens = 1;
end

if nargin<6
  ccolor = 'k'; % contour color
end

if smo(2)>0
  %lms = [0.95]; % p-limits to draw
  lms = [0.62 0.90 0.95]; % p-limits to draw
  %lms = [0.62 0.90]; % p-limits to draw
  %lms = [0.62 0.95]; % p-limits to draw
  %lms = [0.50 0.95]; % p-limits to draw
  %lms = [0.68 0.95 0.99]; % p-limits to draw
  [xo,yo,z,p]=plims2d([x,y],lms,smo(2:end),rho);

  h=gca; hp=findobj(h,'Type','line');

%   set(hp,'MarkerSize',1);
%   set(hp(1),'Color',ccolor);
%   set(hp(1),'Color',[.5 .5 .5]); hold on
hold on
   [c,hc]=contour(xo,yo,z,p);hold on
%
  %get(hc(1))
  for i=1:length(hc); set(hc(i),'LineWidth',1.0); end
  %for i=1:length(hc); set(hc(i),'EdgeColor',ccolor); end
  for i=1:length(hc); set(hc(i),'EdgeColor','k'); end
  %clabel(c,hc)
```

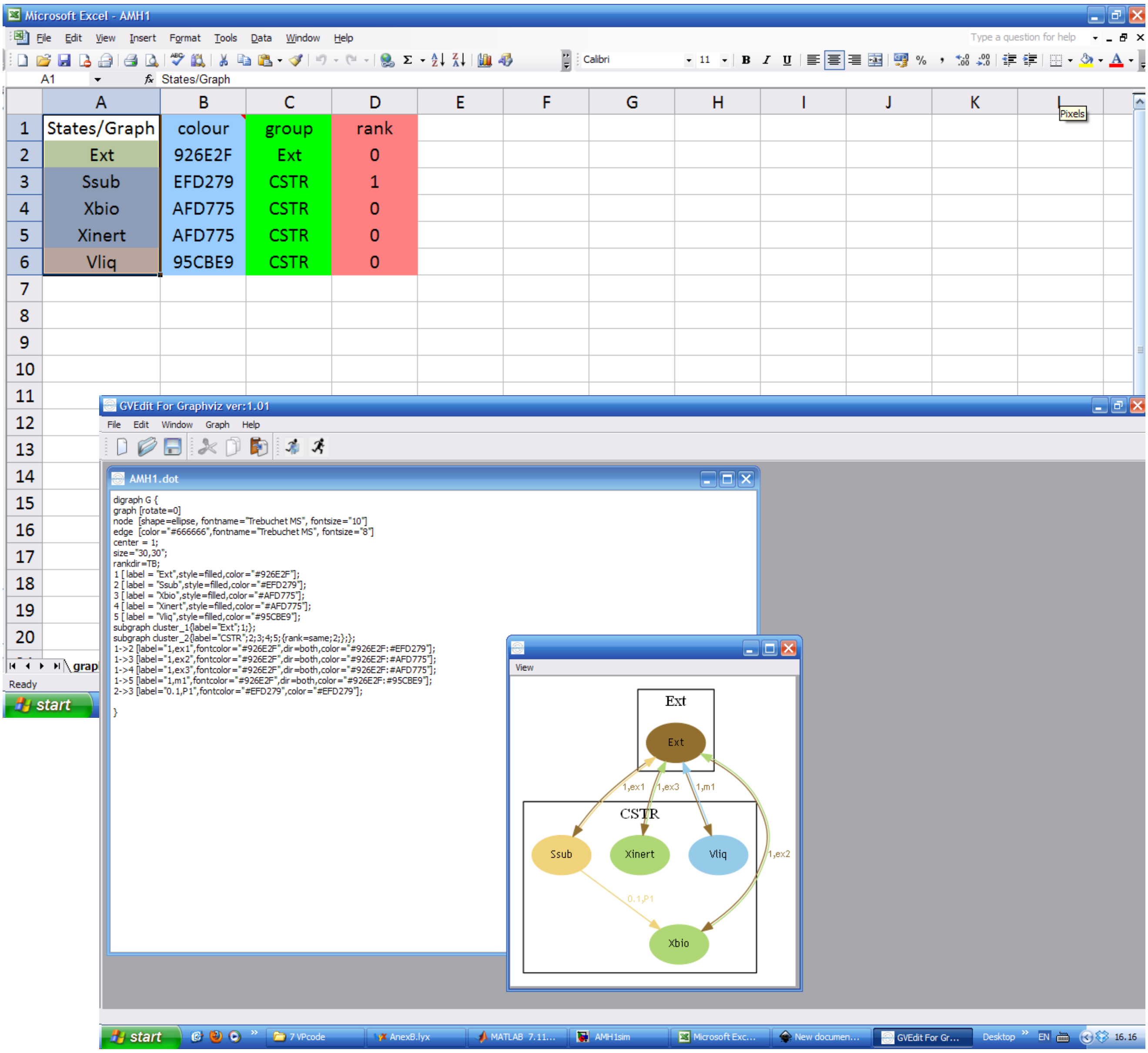
```
end

if (dens&smo(1)>0)
% add marginal densities
[yd1,xd1]=density(x,[],smo(1));
[yd2,xd2]=density(y,[],smo(1));

dscale=0.15; % marginal density scale
%ylim=get(h,'YLim');
ylim=[min(xd2),max(xd2)];
ymax=max(yd1);
%xlim=get(h,'XLim');
xlim=[min(xd1),max(xd1)];
yymax=max(yd2);

y2=(yd1*(ylim(2)-ylim(1))/ymax*dscale + ylim(1));
%plot(xd1,y2,'Color',ccolor,'LineWidth',1.0)

yy2=(yd2*(xlim(2)-xlim(1))/yymax*dscale + xlim(1));
%plot(yy2,xd2,'Color',ccolor,'LineWidth',1.0)

axis([xlim,ylim])
%axis([min(xd1),max(xd1),min(xd2),max(xd2)])
end

hold off
```


MATLAB 7.11.0 (R2010b)
File Edit View Debug Parallel Desktop Window Help
Current Folder: C:\Documents and Settings\Uko\Documenti\My Dropbox\Matlab\VP3\Models
Shortcuts How to Add What's New
Current Folder
« Matlab ▸ VP3 ▸ Models

Name	Type
AMH1.c	C Source
AMH1.dot	Microsoft Word Template
AMH1_f.m	MATLAB Function
AMH1_J.m	MATLAB Function
ADM1co_SAVApHodesim.mdl	Simulink Model File
AMH1sim.mdl	Simulink Model File
ASMAcsim.mdl	Simulink Model File
AMH1.mexw32	File MEXW32
AMH1_SShs.txt	Documento di testo
AMH1_Xlhs.txt	Documento di testo
ADM1co_SAVApHode.xls	Microsoft Excel Worksheet
AMH1.xls	Microsoft Excel Worksheet
ASMAc.xls	Microsoft Excel Worksheet

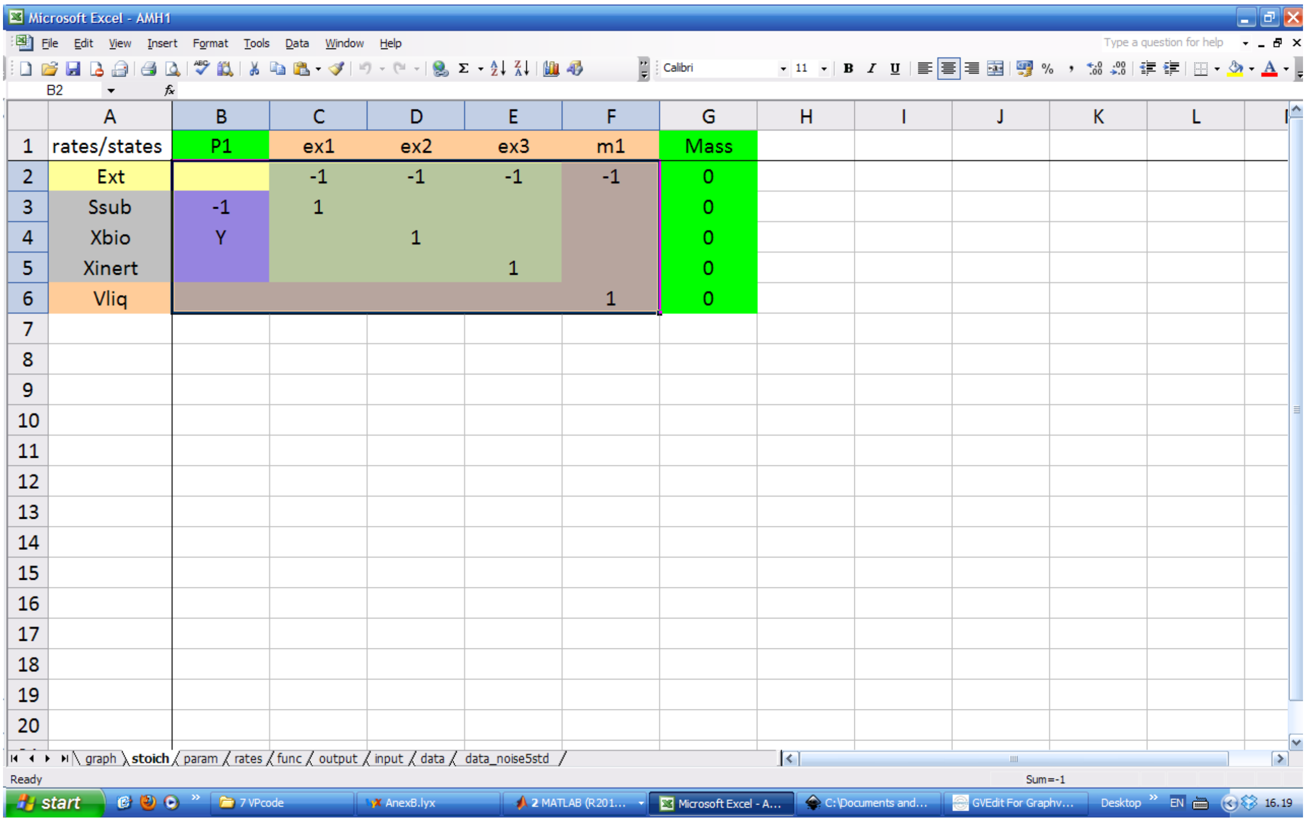
AMH1sim.mdl (Simulink Model File)
Workspace
Stack: Base
Select data to plot
Name
Value
Editor - C:\Documents and Settings\Uko\Documenti\My Dropbox\Matlab\VP3\mainAMH1.m
Stack: Base

```
%% Model info
clear
% Define and describe the model
Code.modelname='AMH1';
Code.infoauthor={'Zivko Juznic-Zonta';['Copyrigth,' date];...
                 'Mollet del Valles, BCN, Spain';'UPC/GIRO-CT'};
Code.infomodel={'Simple modified Haldane model (Andrews, 1968) for AD'};
% Load the proyect from xls
S=loadxl
% Transf
S=sysinf
S=mcmcpa
% Path r                                                          puted)
Adj_s=pa
% Save t
eval(['s

%% Write

% Build
Code=bui
% Write
CmexMode
%MfileMo
```

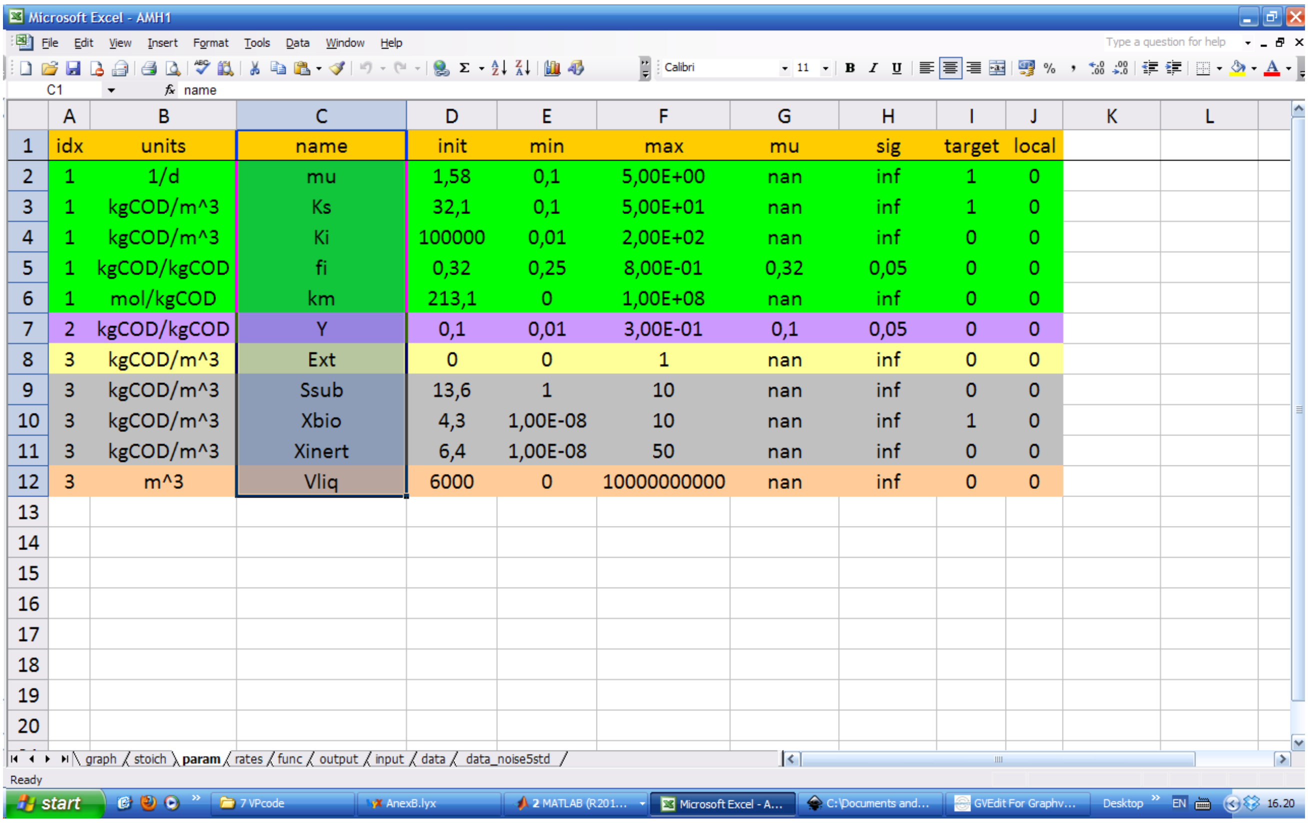


	A	B	C	D
1	States/Graph	colour	group	rank
2	Ext	926E2F	Ext	0
3	Ssub	EFD279	CSTR	1
4	Xbio	AFD775	CSTR	0
5	Xinert	AFD775	CSTR	0
6	Vliq	95CBE9	CSTR	0

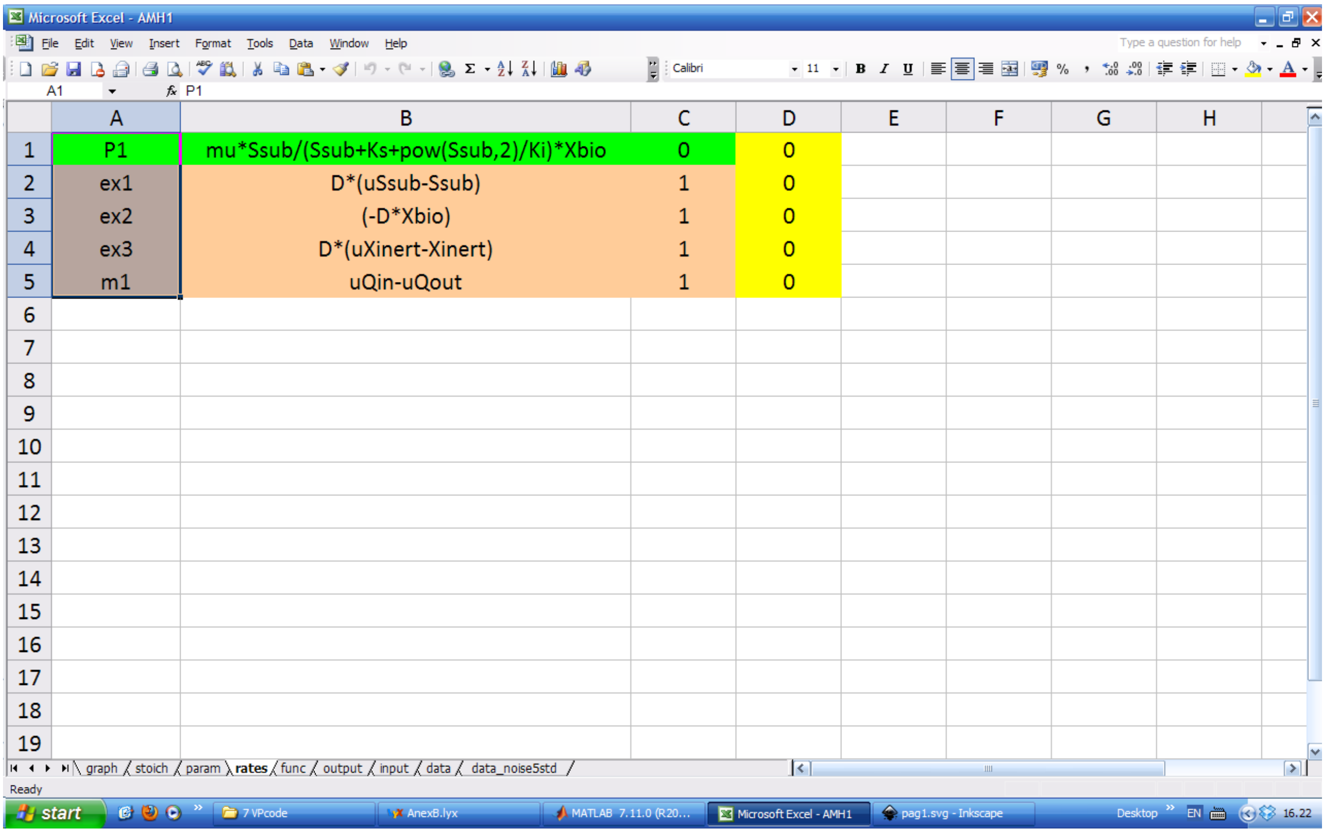
```
digraph G {
graph [rotate=0]
node  [shape=ellipse, fontname="Trebuchet MS", fontsize="10"]
edge  [color="#666666",fontname="Trebuchet MS", fontsize="8"]
center = 1;
size="30,30";
rankdir=TB;
1 [ label = "Ext",style=filled,color="#926E2F"];
2 [ label = "Ssub",style=filled,color="#EFD279"];
3 [ label = "Xbio",style=filled,color="#AFD775"];
4 [ label = "Xinert",style=filled,color="#AFD775"];
5 [ label = "Vliq",style=filled,color="#95CBE9"];
subgraph cluster_1{label="Ext";1;};
subgraph cluster_2{label="CSTR";2;3;4;5;{rank=same;2;};};
1->2 [label="1,ex1",fontcolor="#926E2F",dir=both,color="#926E2F:#EFD279"];
1->3 [label="1,ex2",fontcolor="#926E2F",dir=both,color="#926E2F:#AFD775"];
1->4 [label="1,ex3",fontcolor="#926E2F",dir=both,color="#926E2F:#AFD775"];
1->5 [label="1,m1",fontcolor="#926E2F",dir=both,color="#926E2F:#95CBE9"];
2->3 [label="0.1,P1",fontcolor="#EFD279",color="#EFD279"];

}
```


Microsoft Excel - AMH1
File Edit View Insert Format Tools Data Window Help
B2
graph stoich param rates func output input data data_noise5std
Ready
Sum=-1
start
16.19

rates/states	P1	ex1	ex2	ex3	m1	Mass
Ext		-1	-1	-1	-1	0
Ssub	-1	1				0
Xbio	Y		1			0
Xinert				1		0
Vliq					1	0

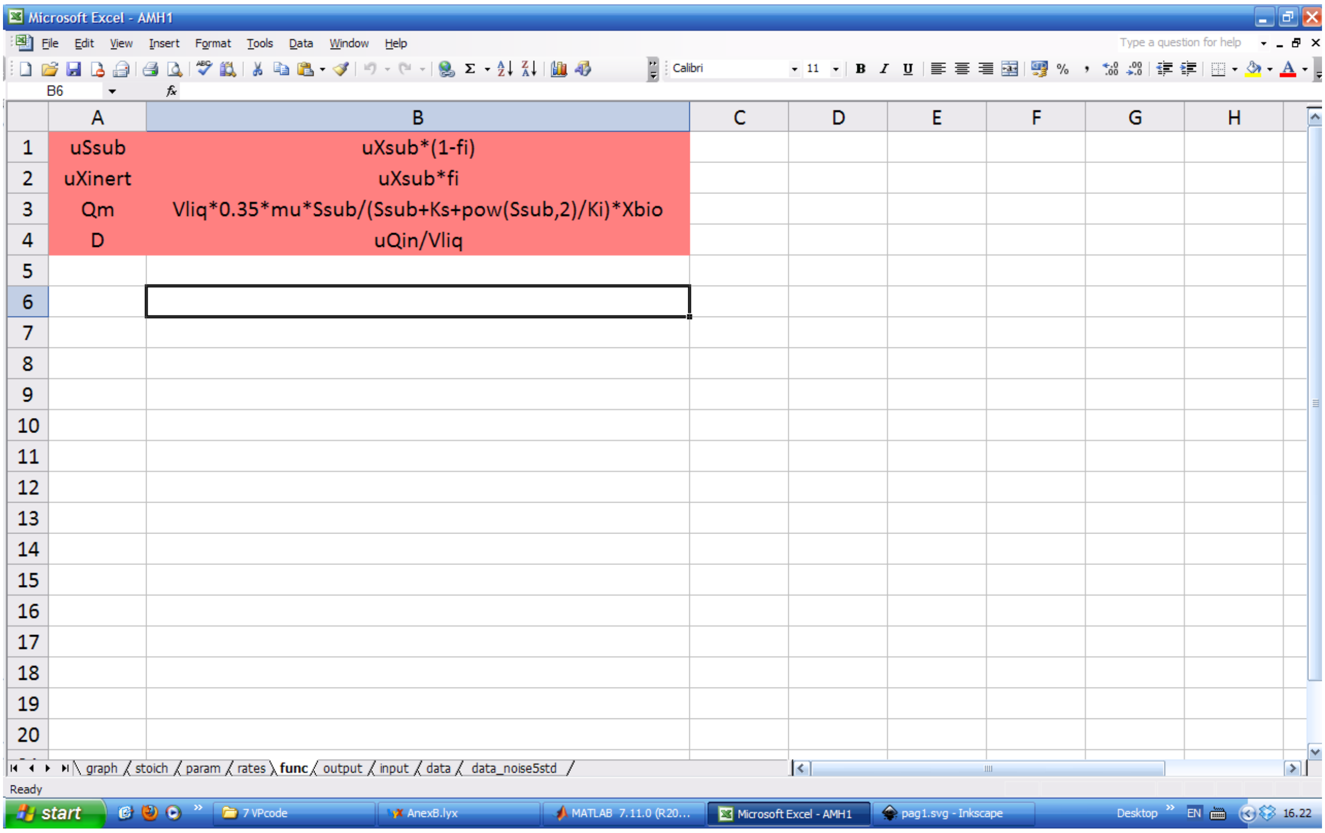

Microsoft Excel - AMH1
File Edit View Insert Format Tools Data Window Help
C1
name
graph stoich param rates func output input data data_noise5std
Ready
start
16.20

idx	units	name	init	min	max	mu	sig	target	local
1	1/d	mu	1,58	0,1	5,00E+00	nan	inf	1	0
1	kgCOD/m^3	Ks	32,1	0,1	5,00E+01	nan	inf	1	0
1	kgCOD/m^3	Ki	100000	0,01	2,00E+02	nan	inf	0	0
1	kgCOD/kgCOD	fi	0,32	0,25	8,00E-01	0,32	0,05	0	0
1	mol/kgCOD	km	213,1	0	1,00E+08	nan	inf	0	0
2	kgCOD/kgCOD	Y	0,1	0,01	3,00E-01	0,1	0,05	0	0
3	kgCOD/m^3	Ext	0	0	1	nan	inf	0	0
3	kgCOD/m^3	Ssub	13,6	1	10	nan	inf	0	0
3	kgCOD/m^3	Xbio	4,3	1,00E-08	10	nan	inf	1	0
3	kgCOD/m^3	Xinert	6,4	1,00E-08	50	nan	inf	0	0
3	m^3	Vliq	6000	0	10000000000	nan	inf	0	0



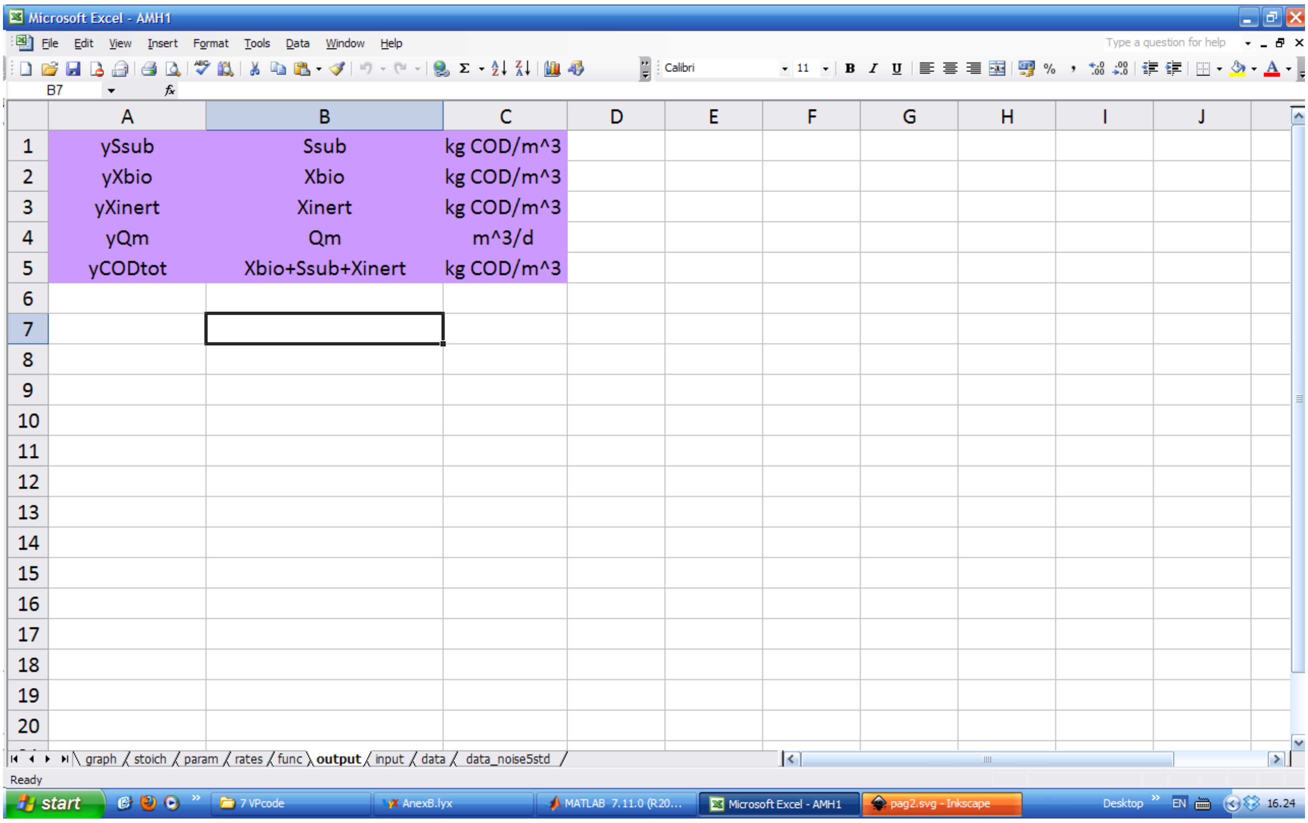
A1 fx P1

	A	B	C	D	E	F	G	H
1	P1	mu*Ssub/(Ssub+Ks+pow(Ssub,2)/Ki)*Xbio	0	0				
2	ex1	D*(uSsub-Ssub)	1	0				
3	ex2	(-D*Xbio)	1	0				
4	ex3	D*(uXinert-Xinert)	1	0				
5	m1	uQin-uQout	1	0				

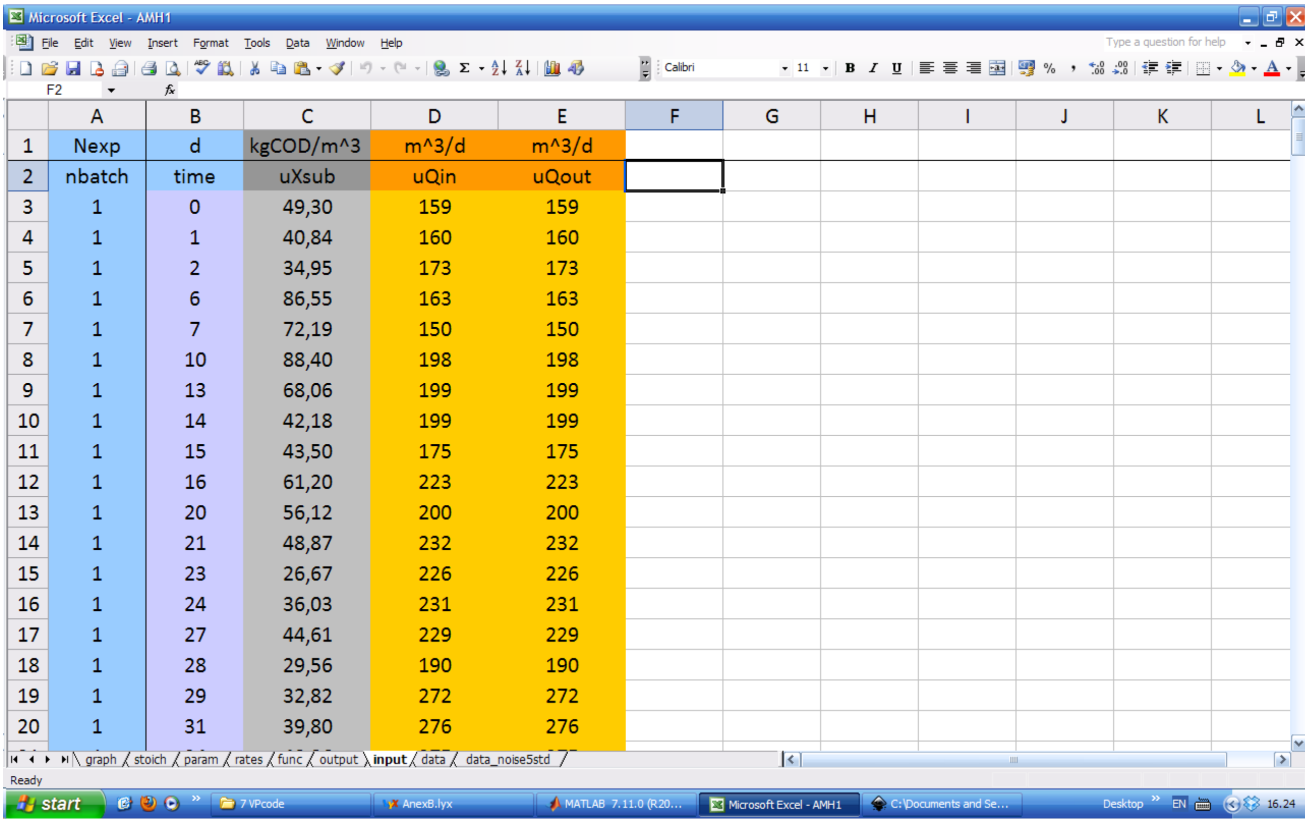


B6 fx

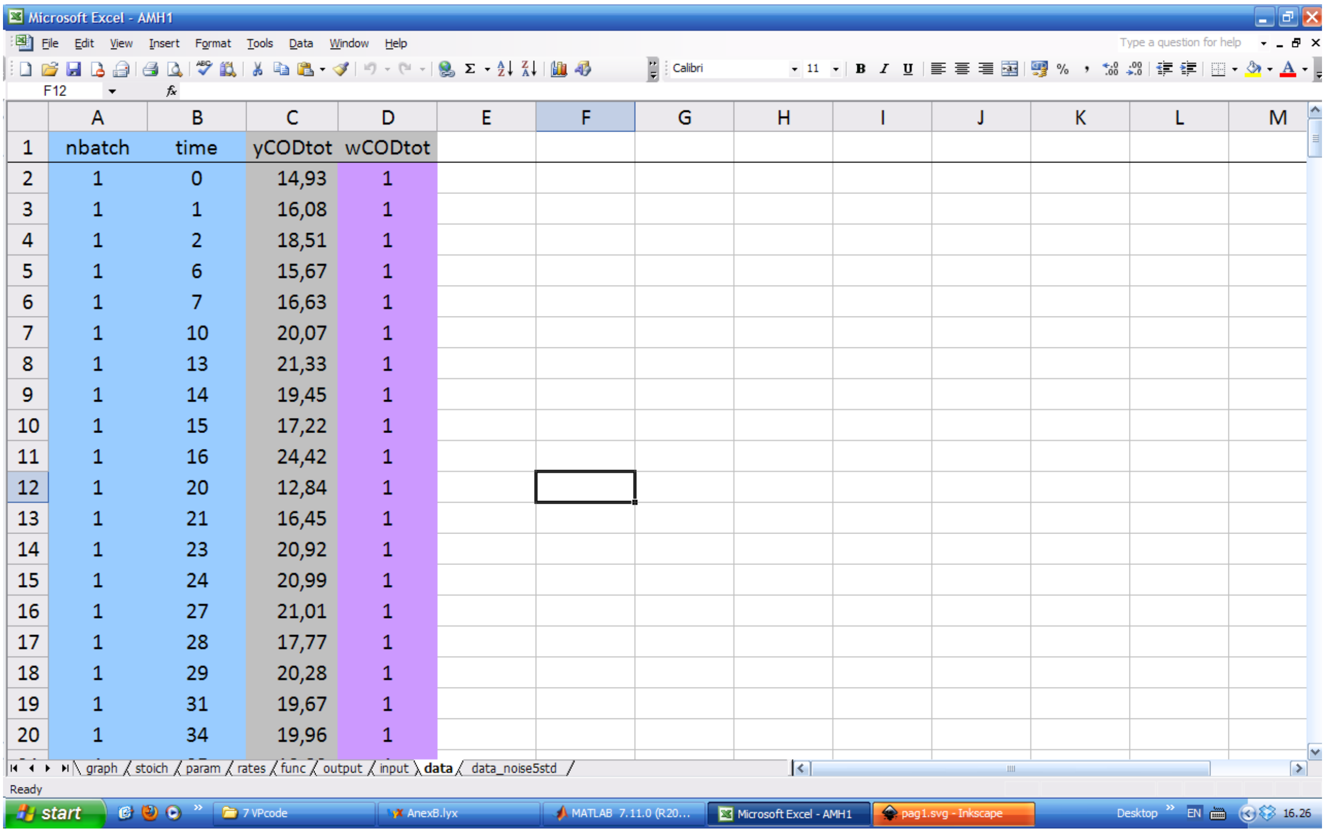
	A	B	C	D	E	F	G	H
1	uSsub	uXsub*(1-fi)						
2	uXinert	uXsub*fi						
3	Qm	Vliq*0.35*mu*Ssub/(Ssub+Ks+pow(Ssub,2)/Ki)*Xbio						
4	D	uQin/Vliq						



	A	B	C
1	ySsub	Ssub	kg COD/m^3
2	yXbio	Xbio	kg COD/m^3
3	yXinert	Xinert	kg COD/m^3
4	yQm	Qm	m^3/d
5	yCODtot	Xbio+Ssub+Xinert	kg COD/m^3



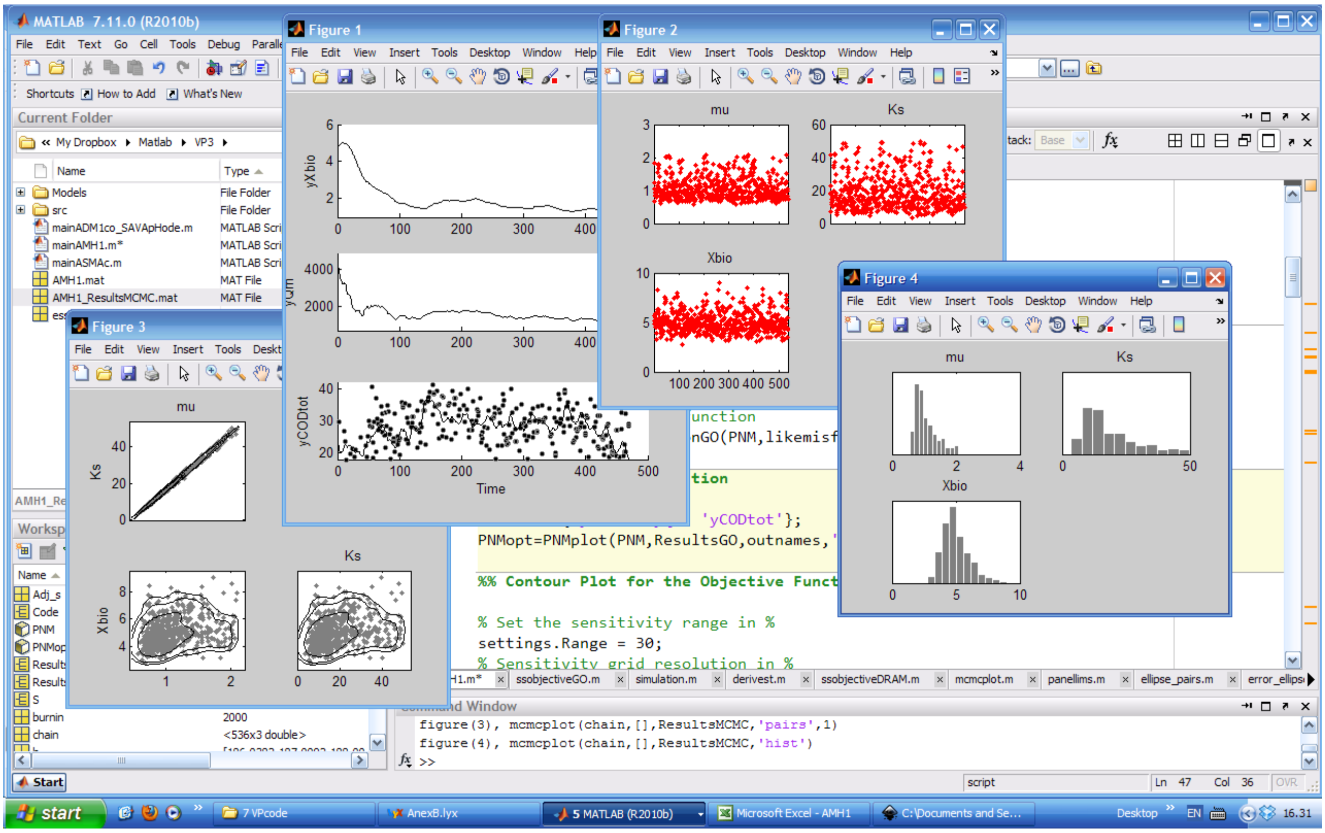
	A	B	C	D	E
1	Nexp	d	kgCOD/m^3	m^3/d	m^3/d
2	nbatch	time	uXsub	uQin	uQout
3	1	0	49,30	159	159
4	1	1	40,84	160	160
5	1	2	34,95	173	173
6	1	6	86,55	163	163
7	1	7	72,19	150	150
8	1	10	88,40	198	198
9	1	13	68,06	199	199
10	1	14	42,18	199	199
11	1	15	43,50	175	175
12	1	16	61,20	223	223
13	1	20	56,12	200	200
14	1	21	48,87	232	232
15	1	23	26,67	226	226
16	1	24	36,03	231	231
17	1	27	44,61	229	229
18	1	28	29,56	190	190
19	1	29	32,82	272	272
20	1	31	39,80	276	276



Microsoft Excel - AMH1

	A	B	C	D
1	nbatch	time	yCODtot	wCODtot
2	1	0	14,93	1
3	1	1	16,08	1
4	1	2	18,51	1
5	1	6	15,67	1
6	1	7	16,63	1
7	1	10	20,07	1
8	1	13	21,33	1
9	1	14	19,45	1
10	1	15	17,22	1
11	1	16	24,42	1
12	1	20	12,84	1
13	1	21	16,45	1
14	1	23	20,92	1
15	1	24	20,99	1
16	1	27	21,01	1
17	1	28	17,77	1
18	1	29	20,28	1
19	1	31	19,67	1
20	1	34	19,96	1

File Edit View Insert Format Tools Data Window Help
F12
graph / stoich / param / rates / func / output / input / data / data_noise5std
Ready
start
7 VPcode
AnexB.lyx
MATLAB 7.11.0 (R20...
Microsoft Excel - AMH1
pag1.svg - Inkscape
Desktop
EN
16.26



```
unction
nGO(PNM,likemisf

tion

 'yCODtot'};
PNMopt=PNMplot(PNM,ResultsGO,outnames,'

%% Contour Plot for the Objective Funct

% Set the sensitivity range in %
settings.Range = 30;
% Sensitivity grid resolution in %
```

```
figure(3), mcmcplot(chain,[],ResultsMCMC,'pairs',1)
figure(4), mcmcplot(chain,[],ResultsMCMC,'hist')
>>
```