UNIVERSITAT DE BARCELONA

Exploración y modelización de patrones socioecológicos y tecnoculturales en sociedades preindustriales de zonas áridas afro-eurasiáticas

Una aproximación multidisciplinar desde métodos cuantitativos

Andreas Angourakis

Apéndice A: Apéndices de artículos publicados

Este apéndice recoge todos los apéndices correspondientes a los artículos originales publicados que forman esta tesis.

A.1 Análisis de datos arqueométricos

5.3.3. A.1.1 Apéndices de 'Assessing Hellenistic to nomadic cultural patterns through pottery in ancient Termez, Uzbekistan'

Los apéndices de Martínez Ferreras et al. ([sin fecha]) (sección 3.2) son los siguientes:

- Figure S1: The wares analysed by SEM-EDS to investigate the surface treatments.
- Table S1a: XRF normalised data of the 58 pottery samples and the 7 clayey sediments analysed by WD-XRF on the subcomposition Fe_2O3, Al_2O_3, P_2O_5, TiO_2, MgO, CaO, Na_2O, K_2O, SiO_2 (in %), Ba, Rb, Th, Nb, Pb, Zr, Y, Sr, Ce, Ga, V, Zn, Cu, Ni, Cr (in ppm), and the loss on ignition (LOI) (in %).
- Table S1b: Summary of the Compositional Variation Matrix of the 58 pottery wares analysed.
- Table S1c: Summary of the Compositional Variation Matrix of the 58 vessels and the 7 clayey sediments analysed.

Figura 5.1: The wares analysed by SEM-EDS to investigate the surface treatments. Scale bar: 5 cm. *Figure S1 in the original.*

Cuadro 5.1: **(a)** XRF normalised data of the 58 pottery samples and the 7 clayey sediments analysed by WD-XRF on the subcomposition Fe_2O3, Al_2O_3, P_2O_5, TiO_2, MgO, CaO, Na_2O, K_2O, SiO_2 (in %), Ba, Rb, Th, Nb, Pb, Zr, Y, Sr, Ce, Ga, V, Zn, Cu, Ni, Cr (in ppm), and the loss on ignition (LOI) (in %). *Table S1a in the original.*

	Fe2O3	Al2O3	P2O5	TiO2	MgO	CaO	Na2O	K2O	SiO2	Ba	Rb	Th	Nb	Pb	Zr	Y	Sr	Ce	Ga	V	Zn	Cu	Ni	Cr	LOI
TC-1	6.54	16.22	0.19	0.68	3.51	9.94	1.28	3.69	55.75	547	130	14	15	26	154	25	452	75	19	100	102	39	49	74	2.85
TC-2	6.15	15.83	0.21	0.70	3.32	8.47	1.51	3.51	55.35	561	136	17	18	22	177	27	270	54	19	97	93	35	54	77	2.52
TC-3	6.90	16.14	0.25	0.68	3.91	9.66	1.27	3.55	52.92	578	144	17	16	18	165	28	361	77	22	109	120	40	54	79	1.46
TC-4	5.58	14.63	0.33	0.58	3.35	11.25	1.28	3.48	53.63	513	120	14	15	17	151	24	624	54	18	101	133	26	45	67	4.60
TC-5	6.18	15.46	0.27	0.63	3.25	10.08	0.93	3.40	57.01	536	112	12	14	18	136	23	368	44	16	110	98	29	43	70	2.94
TC-7	5.94	15.31	0.26	0.68	3.05	8.34	1.34	3.42	56.87	547	27	15	16	18	175	27	372	83	18	99	90	32	47	73	5.24
TC-8	5.34	14.21	0.22	0.64	2.59	9.37	1.15	3.30	58.52	502	109	12	15	29	156	24	405	57	16	106	93	26	33	68	2.34
TC-9	5.79	14.58	0.19	0.59	3.66	12.55	2.58	3.61	53.55	473	113	14	13	25	131	21	583	50	15	91	99	30	42	63	3.06
TC-10	5.99	15.38	0.98	0.63	3.39	10.57	1.33	3.61	53.89	531	125	13	14	26	145	25	547	62	18	105	108	30	46	71	3.14
TC-12	5.58	14.32	0.53	0.65	3.01	8.76	1.18	3.40	57.24	531	110	13	16	26	165	25	320	58	17	101	98	29	42	68	4.64
TC-13	6.81	16.67	0.21	0.68	3.57	8.59	1.35	3.76	55.27	545	136	16	15	13	151	25	350	53	19	103	101	37	49	79	4.11
TC-15	5.97	15.49	0.27	0.64	3.09	7.87	1.36	3.39	55.80	531	122	15	15	27	166	24	321	74	18	100	98	32	49	78	3.57
TC-16	6.57	16.39	0.22	0.67	3.60	10.89	1.44	3.21	54.48	579	120	14	15	26	157	25	433	63	19	110	108	36	52	76	1.36
TC-17	5.86	15.36	0.20	0.65	3.24	9.03	1.49	3.35	54.80	484	118	15	15	27	161	24	296	66	17	101	90	31	45	73	3.99
TC-18	6.09	15.24	0.37	0.64	3.20	10.92	1.28	3.51	50.68	571	127	14	16	24	161	27	533	74	19	98	105	35	49	69	6.90

	Fe2O3	Al2O3	P2O5	TiO2	MgO	CaO	Na2O	K2O	SiO2	Ba	Rb	Th	Nb	Pb	Zr	Y	Sr	Ce	Ga	V	Zn	Cu	Ni	Cr	LOI
TC-19	6.58	16.07	0.47	0.67	3.19	8.71	1.08	3.66	56.39	447	110	13	13	25	134	23	304	49	16	113	91	30	40	75	5.47
TC-20	6.72	15.75	0.31	0.65	3.45	10.34	1.20	3.59	55.57	556	132	15	15	23	149	25	487	61	18	92	100	36	48	72	3.72
TC-21	6.12	15.59	0.39	0.64	3.86	12.89	1.91	2.63	52.06	528	95	13	14	24	154	26	509	65	19	101	110	30	50	70	4.52
TC-22	6.17	15.59	0.64	0.66	3.30	10.33	1.40	3.53	54.25	557	124	14	16	24	166	25	569	75	19	87	101	34	48	70	4.08
TP2-1	6.07	14.58	0.21	0.64	3.16	11.59	1.20	2.99	54.58	552	130	16	18	24	162	27	519	65	20	93	86	39	52	78	3.73
TP2-2	6.41	15.94	0.19	0.66	3.57	8.87	1.41	3.22	58.23	553	144	17	17	26	160	27	393	60	20	98	106	41	53	81	1.56
TP2-3	6.28	15.53	0.19	0.69	3.18	9.10	1.06	3.21	56.65	479	137	17	18	30	162	28	419	46	20	107	106	41	51	78	3.77
TP2-4	6.39	16.40	0.19	0.67	3.43	8.15	1.19	3.29	58.21	530	147	19	18	29	154	29	352	59	20	107	108	38	52	87	2.03
TP2-5	5.98	14.92	0.18	0.65	3.27	10.66	1.37	2.90	58.35	546	124	18	18	21	165	27	327	54	19	91	92	38	53	78	1.33
TP2-6	5.72	14.44	0.20	0.64	2.90	9.91	1.16	2.93	55.55	501	126	15	18	25	164	28	401	52	19	101	97	38	46	78	5.59
TP2-7	5.75	15.20	0.19	0.67	3.32	7.50	1.33	3.09	58.09	545	133	18	18	27	163	29	424	69	20	111	114	46	49	83	3.66
TP2-8	5.61	14.30	0.24	0.63	2.96	9.32	1.13	3.01	57.13	491	131	16	18	26	157	26	502	58	19	106	104	39	47	84	4.46
TP2-9	6.49	15.08	0.21	0.70	2.93	6.25	1.10	3.14	57.97	478	125	17	19	27	167	28	439	58	19	117	114	48	45	85	5.56
TP2-10	6.34	15.81	0.22	0.70	3.29	9.57	1.21	3.28	57.73	586	144	19	19	19	164	27	386	57	20	106	101	36	56	83	1.53
TP2-11	5.68	14.52	0.25	0.66	3.10	7.25	1.24	3.05	60.70	558	128	17	18	33	165	27	443	61	19	103	95	53	43	74	2.9
TP2-12	6.24	15.46	0.21	0.67	3.68	11.66	1.19	3.01	55.92	552	131	17	18	24	157	31	317	65	20	96	107	42	53	75	1.4
TP2-13	5.75	14.70	0.19	0.65	2.99	8.14	1.13	3.03	59.29	511	135	18	19	29	178	28	380	54	20	104	100	40	53	80	3.43
TP2-14	5.94	15.41	0.24	0.66	3.25	8.81	1.67	2.88	58.12	616	117	16	17	24	161	29	479	57	21	110	98	38	50	85	2.1

	Fe2O3	Al2O3	P2O5	TiO2	MgO	CaO	Na2O	K2O	SiO2	Ba	Rb	Th	Nb	Pb	Zr	Y	Sr	Ce	Ga	V	Zn	Cu	Ni	Cr	LOI
TP2-15	6.15	15.72	0.21	0.65	3.62	10.48	1.27	3.18	57.23	533	143	16	17	28	161	28	382	52	20	96	100	35	54	82	1.1
TP2-16	5.78	14.98	0.22	0.64	3.70	10.40	1.45	2.84	57.66	526	126	16	17	7	164	28	393	66	19	104	88	37	51	80	1.5
TP2-17	6.14	15.91	0.21	0.67	3.82	9.85	1.38	3.08	57.46	484	139	18	18	8	156	27	346	67	21	108	102	34	55	85	1.10
TP2-18	6.22	15.71	0.21	0.67	3.34	10.30	1.31	3.18	56.56	560	145	18	17	25	159	29	385	66	21	110	105	36	57	83	1.80
TP2-19	5.78	15.29	0.21	0.62	3.82	10.09	1.51	2.84	57.28	499	128	17	17	25	151	28	395	49	20	104	97	34	50	83	1.70
TP2-20	5.52	14.47	0.56	0.61	3.46	10.27	1.38	2.88	56.66	532	131	18	18	19	154	27	460	43	20	102	110	40	49	78	2.9
TP2-21	6.57	16.28	0.19	0.68	3.26	8.39	1.17	3.38	57.72	530	151	17	18	28	159	29	375	56	21	114	102	44	53	82	1.80
TP2-22	6.52	16.03	0.19	0.68	3.73	9.04	1.26	3.15	56.66	538	139	16	18	23	163	29	389	66	22	108	107	44	56	82	2.5
TP2-23	6.32	16.19	0.20	0.68	3.65	8.74	1.44	3.07	57.83	575	135	17	18	21	160	29	349	63	21	108	106	40	53	83	1.07
TP2-24	5.89	14.94	0.25	0.67	3.88	10.73	1.29	3.01	56.62	540	130	17	18	16	166	30	447	63	19	97	102	42	52	76	2.23
TP2-25	6.27	15.79	0.22	0.70	3.98	10.41	1.79	2.42	56.96	470	101	18	18	28	168	28	325	53	21	98	105	37	54	79	1.20
TP2-26	6.56	15.69	0.21	0.67	3.40	9.12	1.05	3.26	53.41	531	137	17	18	32	155	30	436	65	21	117	117	49	54	84	5.83
TP1-1	6.15	16.40	0.24	0.68	3.55	8.14	1.12	3.18	57.19	547	131	21	14	26	150	23	377	78	20	117	106	30	56	86	2.16
TP1-2	6.24	15.51	0.27	0.69	3.01	7.32	0.97	3.03	58.68	586	116	21	14	31	157	23	436	64	17	113	96	37	51	91	3.33
TP1-3	5.72	14.08	0.22	0.62	2.97	9.69	0.89	2.91	54.23	594	111	19	14	30	160	24	489	56	18	104	104	42	49	77	6.93
TP1-4	6.37	15.94	0.23	0.68	3.27	9.10	1.17	3.18	56.52	641	126	20	14	32	153	23	377	62	19	94	83	39	59	86	2.03
TP1-5	6.24	15.58	0.23	0.68	3.39	10.11	1.21	3.11	56.27	597	126	17	15	29	160	24	352	70	18	89	99	46	57	84	0.58
TP1-6	6.08	15.27	0.26	0.67	3.10	8.43	1.25	3.08	57.48	596	121	17	15	31	172	25	420	54	16	99	92	52	50	82	3.36

	Fe2O3	Al2O3	P2O5	TiO2	MgO	CaO	Na2O	K2O	SiO2	Ba	Rb	Th	Nb	Pb	Zr	Y	Sr	Ce	Ga	V	Zn	Cu	Ni	Cr	LOI
TP1-7	6.51	16.42	0.21	0.70	3.39	8.29	1.40	3.30	57.96	633	129	23	14	23	163	24	357	51	20	102	109	43	54	83	1.23
TP1-8	6.07	15.29	0.21	0.67	3.16	9.74	1.07	3.13	57.85	662	119	22	14	35	155	22	366	52	19	96	97	45	55	76	2.06
TP1-9	6.40	15.26	0.47	0.65	3.28	7.28	0.99	3.13	58.17	736	122	21	14	33	150	22	460	69	19	116	102	49	56	90	3.56
TP1-10	6.03	15.32	0.32	0.67	3.04	7.71	1.09	3.14	57.88	680	123	23	15	36	159	22	436	70	19	113	101	43	56	87	3.53
TTT-1	6.39	15.97	0.19	0.67	3.39	8.67	1.47	3.43	56.81	671	132	15	16	28	157	28	451	54	18	109	94	32	50	81	1.63
TTT-2	6.01	15.10	0.21	0.66	3.54	9.50	1.91	3.34	57.44	640	127	15	15	30	155	25	452	51	17	87	92	36	49	73	0.83
TTT-3	5.50	14.33	0.24	0.62	3.11	8.77	1.48	3.15	56.79	531	113	14	15	22	162	24	412	50	16	101	95	25	44	74	3.52
G-1	5.33	13.24	0.19	0.57	3.63	12.37	0.71	2.70	47.70	545	122	16	16	24	130	24	253	51	16	104	93	31	43	70	14.8
G-2	4.77	11.27	0.15	0.58	2.37	13.01	0.83	2.38	50.98	393	110	16	17	26	167	22	187	47	14	82	76	30	36	61	13.6
G-3	3.64	9.80	0.12	0.45	1.82	16.38	1.27	2.12	49.61	490	83	12	9	21	131	19	223	47	12	64	52	16	26	54	15.0
G-4	6.72	14.46	0.12	0.63	3.28	11.25	1.05	3.12	46.06	433	120	14	11	58	130	24	218	66	18	120	110	41	50	69	13.6
G-5	4.29	12.22	0.17	0.62	2.32	6.63	2.37	2.38	61.18	495	84	15	12	21	222	27	188	65	15	84	60	16	22	72	7.1
G-6	5.39	13.22	0.16	0.65	2.74	8.97	1.99	2.82	51.51	391	109	14	13	27	151	23	191	51	17	95	90	29	36	61	12.9
G-7	6.20	14.55	0.16	0.68	2.84	8.24	1.38	3.12	51.83	420	126	15	13	28	149	24	152	59	19	112	105	39	43	69	10.9

Cuadro 5.2: **(b)** Summary of the Compositional Variation Matrix of the 58 pottery wares analysed. *Table S1b in the original.*

	Fe2O3	Al2O3	P2O5	TiO2	MgO	CaO	Na2O	K2O	SiO2	Ba	Rb	Th	Nb	Pb	Zr	Y	Sr	Ce	Ga	V	Zn	Cu	Ni	Cr
τ_i	0.573	0.543	3.443	0.541	0.679	1.487	1.423	0.712	0.590	0.722	0.652	0.945	0.832	2.756	0.633	0.666	1.420	0.952	0.617	0.667	0.664	1.125	0.696	0.611
tv/τ_i	0.870	0.919	0.145	0.923	0.735	0.336	0.351	0.701	0.845	0.690	0.766	0.528	0.600	0.181	0.788	0.750	0.351	0.524	0.809	0.748	0.751	0.444	0.717	0.817
$r_{v,\tau}$	0.996	0.998	0.749	0.998	0.987	0.957	0.946	0.984	0.995	0.992	0.995	0.965	0.988	0.946	0.991	0.989	0.866	0.997	0.994	0.993	0.993	0.946	0.987	0.990
tv	0.499																							

Cuadro 5.3: **(c)** Summary of the Compositional Variation Matrix of the 58 vessels and the 7 clayey sediments analysed. *Table S1c in the original.*

	Fe2O3	Al2O3	P2O5	TiO2	MgO	CaO	Na2O	K2O	SiO2	Ba	Rb	Th	Nb	Pb	Zr	Y	Sr	Ce	Ga	V	Zn	Cu	Ni	Cr
τ_i	0.495	0.473	3.296	0.477	0.578	0.957	1.240	0.657	0.495	0.622	0.594	0.889	0.711	2.411	0.521	0.610	1.124	0.896	0.558	0.580	0.569	0.997	0.613	0.554
tv/τ_i	0.880	0.921	0.132	0.913	0.754	0.455	0.351	0.664	0.880	0.701	0.734	0.490	0.613	0.181	0.836	0.715	0.388	0.486	0.781	0.751	0.766	0.437	0.711	0.786
$r_{v,t\tau}$	0.998	0.998	0.650	0.997	0.981	0.930	0.957	0.980	0.998	0.986	0.995	0.956	0.987	0.913	0.997	0.989	0.933	0.993	0.992	0.995	0.994	0.945	0.985	0.985
tv	0.436																							

5.3.4. A.1.2 Apéndices de 'Presenting multivariate statistical protocols in R using wine Roman amphorae productions in Catalonia, Spain'

Los apéndices de Angourakis et al. (2018) (sección 3.3) son los siguientes:

- Appendix A: Petrographic variables
- Appendix B: Dissimilarity formulae
- Appendix C: Using cerUB for exploring wine Roman amphorae
- Appendix D: Scripts
- Appendix E: Animated GIFs *(no incluido en el documento de la tesis)*

Se mantienen versiones actualizadas de todos los apéndices en repositorio público (https://github.com/Andros-Spica/cerUB_tutorial), incluyendo una versión en línea del tutorial (Appendix C), creada con el paquete 'bookdown' y accesible directamente a través del enlace: https://andros-spica.github.io/cerUB_tutorial.

Appendix A: Petrographic variables

Variable name	Variable code	Provenance related	Values
INCLUS_DISTRIB	I1	no	poorly, poorly to moderately, moderately, moderately to well, well, none
INCLUS_ORIENT	I2	no	unparallel, slightly parallel, parallel, none
TEMP	F1	no	unfired, 700-800oC, 800-900oC, 900-1000oC, 1000-1100oC
ATM	F2	no	reducing, reducing to oxidising, oxidising, indeterminate; in this data set all are oxidising
POST_ATM	F3	no	reducing, reducing to oxidising, oxidising, indeterminate; in this data set all are either oxidising or indeterminate
VOID_OVERALL	V1	no	none, very few, few, common, abundant, very abundant
VOID_VESIC_MEGA	V2	no	For all 'VOID_X_Y' variables: none, few, frequent, predominant
VOID_VESIC_MACRO	V3	no	...
VOID_VESIC_MESO	V4	no	...
VOID_VESIC_MICRO	V5	no	...
VOID_VUGH_MEGA	V6	no	...
VOID_VUGH_MACRO	V7	no	...
VOID_VUGH_MESO	V8	no	...
VOID_VUGH_MICRO	V9	no	...
VOID_CHAN_MEGA	V10	no	...
VOID_CHAN_MACRO	V11	no	...
VOID_CHAN_MESO	V12	no	...
VOID_CHAN_MICRO	V13	no	...
VOID_PLAN_MEGA	V14	no	...
VOID_PLAN_MACRO	V15	no	...
VOID_PLAN_MESO	V16	no	...
VOID_PLAN_MICRO	V17	no	...
COAR_FREQ	L1	no	none, very few, few, common, abundant, very abundant
COAR_GRAINSIZE	L2	no	none, very fine, very fine to fine, fine, fine to medium, medium, medium to coarse, coarse, coarse to very coarse, very coarse
COAR_ROUNDNESS	L3	yes	angular, angular to subangular, subangular, subangular to subrounded, subrounded, subrounded to rounded, rounded, none
COAR_FORM	L4	yes	elongate, elongate to equidimensional, equidimensional, equidimensional to laminar, laminar, none
COAR_SPACING	L5	no	single-spaced, single to double-spaced, double-spaced, double to open-spaced, open-spaced, none
COAR_SORTING	L6	no	poorly-sorted, poorly to moderately-sorted, moderately-sorted, moderately to well-sorted, well-sorted, none

Variable name	Variable code	Provenance related	Values
COAR_R_GRANIT	L7	yes	For all 'COAR_R_X' variables: none, few, common, frequent, dominant, predominant
COAR_R_RHYOL	L8	yes	...
COAR_R_DIOR	L9	yes	...
COAR_R_DAC_AND	L10	yes	...
COAR_R_GABBRO	L11	yes	...
COAR_R_BASALT	L12	yes	...
COAR_R_SYEN	L13	yes	...
COAR_R_TRACHY	L14	yes	...
COAR_R_CONGBREC	L15	yes	...
COAR_R_QTZSANDST	L16	yes	...
COAR_R_FELDSANDST	L17	yes	...
COAR_R_LITSANDST	L18	yes	...
COAR_R_CASILTST	L19	yes	...
COAR_R_FESILTST	L20	yes	...
COAR_R_CAMUDST	L21	yes	...
COAR_R_FEMUDST	L22	yes	...
COAR_R_CLAYST	L23	yes	...
COAR_R_LIMEST	L24	yes	...
COAR_R_CALS	L25	yes	...
COAR_R_DOLOM	L26	yes	...
COAR_R_CALM	L27	yes	...
COAR_R_SPELEO	L28	yes	...
COAR_R_CAL.FOS	L29	yes	...
COAR_R_BIVAL	L30	yes	...
COAR_R_TRAV	L31	yes	...
COAR_R_EVAP	L32	yes	...
COAR_R_CHERT	L33	yes	...
COAR_R_RADIO	L34	yes	...
COAR_R_SLATE	L35	yes	...
COAR_R_PHYLL	L36	yes	...
COAR_R_SCHIST	L37	yes	...
COAR_R_GNEISS	L38	yes	...
COAR_R_QUARTZ	L39	yes	...
COAR_R_MARBLE	L40	yes	...
COAR_R_AMP	L41	yes	...
COAR_R_SERP	L42	yes	...
COAR_C_QTZ	L43	yes	For all 'COAR_C_X' variables: none, few, common, frequent, dominant, predominant
COAR_C_PL	L44	yes	...
COAR_C_KFS	L45	yes	...
COAR_C_SA	L46	yes	...
COAR_C_MS	L47	yes	...
COAR_C_BT	L48	yes	...
COAR_C_SRP	L49	yes	...
COAR_C_OP	L50	yes	...
COAR_C_RT	L51	yes	...
COAR_C_SPL	L52	yes	...
COAR_C_EP	L53	yes	...
COAR_C_AM	L54	yes	...
COAR_C_CPX	L55	yes	...

Variable name	Variable code	Provenance related	Values
COAR_C_OPX	L56	yes	...
COAR_C_OL	L57	yes	...
COAR_C_GRT	L58	yes	...
COAR_C_SIL	L59	yes	...
COAR_C_ST	L60	yes	...
COAR_C_TTN	L61	yes	...
COAR_C_ZRN	L62	yes	...
COAR_C_AP	L63	yes	...
COAR_C_PY	L64	yes	...
FINE_FREQ	S1	no	none, very few, few, common, abundant, very abundant
FINE_GRAINSIZE	S2	no	none, very fine silt, very fine to fine silt, fine silt, fine to medium silt, medium silt, medium to coarse silt, coarse silt, coarse silt to very fine sand
FINE_FORM	S3	yes	elongate, elongate to equidimensional, equidimensional, equidimensional to laminar, laminar, none
FINE_C_CAL	S4	yes	For all 'FINE_C_X' variables: none, few, frequent, predominant
FINE_C_CALFOS	S5	yes	...
FINE_C_QTZ	S6	yes	...
FINE_C_PL	S7	yes	...
FINE_C_KFS	S8	yes	...
FINE_C_SA	S9	yes	...
FINE_C_MS	S10	yes	...
FINE_C_BT	S11	yes	...
FINE_C_SRP	S12	yes	...
FINE_C_OP	S13	yes	...
FINE_C_RT	S14	yes	...
FINE_C_EP	S15	yes	...
FINE_C_AM	S16	yes	...
FINE_C_CPX	S17	yes	...
FINE_C_OPX	S18	yes	...
FINE_C_OL	S19	yes	...
FINE_C_GRT	S20	yes	...
FINE_C_ZRN	S21	yes	...

Appendix B: Dissimilarity formulae

Dissimilarity in ordinal variables (NI/RRD)

The neighbor interchange (NI) approach defines the similarity between two observations i and j for variable k (s_{ijk}) as:

$$s_{ijk} = \begin{cases} 1 & \text{if } r_{ik} = r_{jk} \\ 1 - \frac{|r_{ik} - r_{jk}| - \frac{T_{ik}-1}{2} - \frac{T_{jk}-1}{2}}{(\max r_k - \min r_k - \frac{T_{(k,\max)}-1}{2} - \frac{T_{(k,\min)}-1}{2}} \in [0,1] & \text{otherwise} \end{cases} \quad (1)$$

Where r_{ik} is the rank score of observation i for variable k; T_{ik} is the number of observations which have the same rank score for variable k as object i (including i); $T_{(k,max)}$ and $T_{(k,min)}$ are the number of objects which have, respectively, the maximum and minimum rank for variable k. As explained by Podani (1999), the numerator is the minimum number of steps (interchanges of neighbouring values in the ordering) between value x_{ik} and x_{jk}. However, Podani offers a second metricised approach, the relative rank difference (RRD), which is a simplification of the NI formula, but still respects the ordinal nature of variables:

$$s_{ijk} = 1 - \frac{|r_{ik} - r_{jk}|}{\max r_k - \min r_k} \in [0,1] \quad (2)$$

Extended Gower distance

According to Pavoine et al. (2009)'s extended Gower distance, the global distance between two observations i and j (D_{ij}) is the square root of the average squared distances between n observations for n variables; where variables weights (w_{ijk}) and the distance function itself (d_{ijk}) vary depending on variables' nature:

$$D_{ij} = \sqrt{\frac{\sum_{k=1}^{n} d_{ijk}^2 \delta_{ijk} w_k}{\sum_{k=1}^{n} \delta_{ijk} w_k}} \in [0,1] \quad (3)$$

Following the philosophy of Gower's general coefficient of similarity (1971), from which this distance derives, δ_{ijk} is equal to 1, unless the value of the kth variable is missing for one or both observations i and j, in which case it is equal to 0 (i.e., invalidates the comparison). Concerning ordinal variables, we took advantage of this feature to consider specific values in certain variables to add nothing to this distance-i.e., they do not differentiate from any other value. For instance, in most variables regarding voids and inclusions, "none" values count as the minimum category of the ordinal gradient representing frequency, assuming that "none" is more similar to "few" than to "dominant". Exceptionally, when "none" equals the complete absence of a measured trait (e.g., in inclusions distribution and grain form), it is considered a missing value. Where the frequency of missing values causes distances to violate the metric axioms of Euclidean space, Lingoes transformation is applied (Lingoes 1971; after Pavoine et al. 2009).

Gower, J. C. (1971). A General Coefficient of Similarity and Some of Its Properties. *Biometrics* **27**: 857–871.

Lingoes, J. C. (1971). Some boundary conditions for a monotone analysis of symmetric matrices. *Psychometrika* **36**: 195–203.

Pavoine, S., Vallet, J., Dufour, A.-B., Gachet, S. and Daniel, H. (2009). On the challenge of treating various types of variables: application for improving the measurement of functional diversity. *Oikos* **118**: 391–402.

Podani, J. (1999). Extending Gower's General Coefficient of Similarity to Ordinal Characters on JSTOR. *Taxon* **48**: 331–340.

Appendix C: Using cerUB for exploring wine Roman amphorae

Andreas Angourakis and Verònica Martínez Ferreras

2018-02-16

Abstract

This document shows how to use *cerUB* protocols to explore petrographic and compositional data. As an example, we provide the 'amphorae' dataset concerning wine Roman amphorae from sites in Catalonia, NE Spain.

Contents

1 Installing cerUB

There are two options for installing the *cerUB* package:

a. Downloading the latest release version from Zenodo.org and installing it in RStudio (Tools > Install Packages... > Install from: Package Archive File).

   Angourakis, Andreas, & Martínez Ferreras, Verònica. (2017, September 23). cerUB - Protocols for exploring archaeometric data (R package). Zenodo. http://doi.org/10.5281/zenodo.975451

b. Installing the latest development version directly from GitHub (Andros-Spica/cerUB), using the *devtools* package:

```
# this will install devtools package, if not installed already
if (!require("devtools"))
    install.packages("devtools")

devtools::install_github("Andros-Spica/cerUB")
```

```
#> Warning: package 'devtools' was built under R version 3.4.3
```

The second option is recommended, because it is a faster way to install and update packages that are not in CRAN.

The same options are available for installing the *biplot2d3d* package, which we use here to plot protocols results.

> Andreas Angourakis. (2017, September 20). biplot2d3d - an R package for generating highly-customizable biplots. Zenodo. http://doi.org/10.5281/zenodo.897603

```
devtools::install_github("Andros-Spica/biplot2d3d")
```

Any other package required by these two packages (*ade4*, *rgl*, etc.) should be automatically installed.

< back to index

2 Initial procedures

Load *cerUB* package.

```
library(cerUB)
```

2.1 Set directories for saving data

Set a list containing directories for easiness of reference:

```
directories <- list(
  # where you raw data is
  data = "data",
  # where to save transformed compositional data (CoDa)
  transCoDa = "transformed_CoDa",
  # where to save files concerning protocol workflow
  prot1 = "Protocol_1_geochemical_data",
  prot2 = "Protocol_2_petrographic_data",
  prot3 = "Protocol_3_geochemical_and_petrographic_data",
  prot4 = "Protocol_4_provenance_data",
  prot4_Shipwreck = "Protocols_4_provenance_data_with_shipwrecks"
)
```

Create the respective folders in the current R session working directory, if they do not exist:

```
lapply(directories, dir.create, showWarnings = FALSE)
```

2.2 Read data

Load the amphorae dataset:

```
data(amphorae)
```

Or, alternatively, your own dataset (e.g., CSV):

```
dt <- cbind(read.csv(paste(directories$data,
                           "petrographic_data.csv",sep="/"),
                     # assuming the first column contains row names
                     row.names=1),
            read.csv(paste(directories$data,
                           "geochemical_data.csv",sep="/"),
```

```
                        # assuming the first column contains row names
                        row.names=1))
```

Note that if you use your own dataset you must replace all references to "amphorae" with your data frame name (e.g. "dt").

2.3 Codify petrographical variables

Create a two-column data frame containing the original names of petrographic variables and their respective codes:

```
varCode <- code_variables(amphorae)
```

Petrographic variables must be named following *cerUB* naming system. Consult the documentation on the "amphorae" dataset by running:

```
?amphorae
```

The result of the **code_variables** function is used in the **apply_ordination** function for protocols "2a", "2b", "3", and "4" (i.e., whenever there are ordinal input variables). The data frame containing the codification ('varCode') is attached as "ordination_object$variable_tags" to the resulting ordination object.

Variable name	Variable code
Site_Name	Site_Name
LOCATION_SITE_INITIALS	LOCATION_SITE_INITIALS
CHARAC	CHARAC
FabricGroup	FabricGroup
ChemReferenceGroup	ChemReferenceGroup
INCLUS_DISTRIB	I1
INCLUS_ORIENT	I2
TEMP	F1
ATM	F2
POST_ATM	F3
VOID_OVERALL	V1
VOID_VESIC_MEGA	V2
...	...

2.4 Clean and format data

Cleaning and format procedures, including coercing variables as numeric or factor, excluding columns (constants, perturbed, unreliable) and rows (incomplete data, outliers).

```
cleanAmphorae <- clean_and_format(
  amphorae,
  completion_variable = c(
    # The variable with completion info
    "CHARAC",
    # the value indicating completion
    "complete"
  ),
  categorical_columns = 1:112,
  numerical_columns = 113:ncol(amphorae),
  # values converted to NA
  as_na = c("NULL", "indeterminate", "unfired"),
  # method for replacing NAs
  method = NULL,
  # don't use the following variables
  columns_to_exclude = c("VOID_VESIC_MEGA", "VOID_VUGH_MEGA",
                         "VOID_CHAN_MEGA", "VOID_PLAN_MEGA",
                         "COAR_R_DAC_AND", "COAR_R_EVAP",
                         "COAR_R_CONGBREC", "COAR_R_SERP",
                         "COAR_C_SPL", "COAR_C_OPX",
                         "COAR_C_OL", "COAR_C_SIL",
                         "COAR_C_ST", "COAR_C_ZRN",
                         "COAR_C_PY", "FINE_C_OPX",
                         "FINE_C_ZRN"),
  # don't use the following observations
  # (Italic amphorae from Port Vendres 4)
  rows_to_exclude = c("PV4033", # PV4-IND4
                      "PV4017", # PV4-CAMP
                      # PV4-ITT
                      "PV4021", "PV4023", "PV4024",
                      "PV4025", "PV4035", "PV4037",
                      # PV4-NAP
                      "PV4022", "PV4026", "PV4027",
                      "PV4028", "PV4029", "PV4030",
                      "PV4036")
)
```

	Variables	Observations
amphorae	138	238
cleanAmphorae	91	223

2.5 Subsetting criteria

Build vector indicating whether each observation is from a shipwreck:

```
isShipwreck <-
  cleanAmphorae$Site_Name=="Cap del Vol" |
  cleanAmphorae$Site_Name=="Ullastres I" |
  cleanAmphorae$Site_Name=="Port-Vendres 4"
```

Workshops	Shipwrecks
175	48

Build vectors indicating provenance group and whether observations are true outliers (IND, observations with no group assigned). Also, reformat "FabricGroup" and "ChemReferenceGroup", so true outliers are singled out separately and not as a extra group.

```
ProvenanceGroup <- c()
isTrueIND <- c()

# coerce the original group variables (factors) into character vectors
# so we can use stringr package to operate on them.
cleanAmphorae$FabricGroup <-
  as.character(cleanAmphorae$FabricGroup)
cleanAmphorae$ChemReferenceGroup <-
  as.character(cleanAmphorae$ChemReferenceGroup)

for (i in 1:nrow(cleanAmphorae)){
  groupChem <-
    stringr::str_split(cleanAmphorae$ChemReferenceGroup[i], "-")[[1]]
  groupFabric <-
    stringr::str_split(cleanAmphorae$FabricGroup[i], "-")[[1]]
  group <- ""
  isATrueInd <- FALSE

  if (groupChem[2] == "IND" || groupFabric[2] == "IND") {
```

```
    group <- cleanAmphorae$ChemReferenceGroup[i]
    if (!isShipwreck[i]) isATrueInd <- TRUE
    index <- 1
    for (j in 1:length(ProvenanceGroup)){
      if (ProvenanceGroup[j] == paste(group, index, sep = ""))
        index <- index + 1
    }
    group <- paste(group, index, sep = "")
    cleanAmphorae$ChemReferenceGroup[i] <- group
    cleanAmphorae$FabricGroup[i] <- group
  }
  else {
    if (groupChem[1] == "ULL" ||
        groupChem[1] == "PV4" ||
        groupChem[1] == "CDV") {
      group <- cleanAmphorae$ChemReferenceGroup[i]
    }
    else if (groupChem[1] == groupFabric[1]){
      group <- groupChem[1]
    }
  }
  ProvenanceGroup <- c(ProvenanceGroup, group[1])
  isTrueIND <- c(isTrueIND, isATrueInd)
}
```

Assigned	Outliers
205	18

2.6 Organizing groups

Build lists of named group factors for easiness of reference.

First, create a list aiming to define workshops productions, so no shipwrecks:

```
factor_list <-
  list(
    Site = factor(cleanAmphorae$Site_Name[!isShipwreck]),
    FabricGroup = factor(cleanAmphorae$FabricGroup[!isShipwreck]),
    ChemGroup = factor(cleanAmphorae$ChemReferenceGroup[!isShipwreck]),
    ProvGroup = factor(ProvenanceGroup[!isShipwreck])
  )
```

	Site	FabricGroup	ChemGroup	ProvGroup
ACM001	Ca L'Arnau-Can Pau Ferrer	ACM-2	ACM-C	ACM
ACM097	Ca L'Arnau-Can Pau Ferrer	ACM-1	ACM-A	ACM
BIF006	BDN-Pompeu Fabra	BIF-1	BIF-2	BIF
BIF048	BDN-Pompeu Fabra	BIF-1	BIF-3	BIF
CRC002	Cal Ros de les Cabres	CRC-1	CRC-1	CRC
ELV002	El Vilarenc	ELV-1	ELV-1	ELV
ELV051	El Vilarenc	ELV-1	ELV-2	ELV
FEU009	Can Feu	FEU-IND1	FEU-IND1	FEU-IND1
LLA010	Llafranc	LLA-1	LLA-A2	LLA
MOR018	El More	MOR-2	MOR-3	MOR
SAL025	La Salut	SAL-2	SAL-1	SAL
SBL045	Sant Boi Historic Centre	SBL-2	SBL-2	SBL

Then, create a second list, aiming to assign shipwreck observations to workshop productions, so with shipwreck samples but no true outliers:

```
factor_list_Shipwreck <-
  list(
    Site = factor(cleanAmphorae$Site_Name[!isTrueIND]),
    FabricGroup = factor(cleanAmphorae$FabricGroup[!isTrueIND]),
    ChemGroup = factor(cleanAmphorae$ChemReferenceGroup[!isTrueIND]),
    ProvGroup = factor(ProvenanceGroup[!isTrueIND])
  )
```

2.7 Helper objects for plotting

Build lists of named point types vectors for easiness of reference.

Create point type vectors for the whole dataset:

```
labels_code <- as.character(row.names(cleanAmphorae)) # using row names
labels_cross <- rep("+", nrow(cleanAmphorae)) # using +
labels_x <- rep(4, nrow(cleanAmphorae)) # using pch code
labels_point <- rep(20, nrow(cleanAmphorae)) # using pch code
```

Create a list aiming to define workshops productions:

```
labels_list <- list(
  Code = labels_code[!isShipwreck],
  Cross = labels_cross[!isShipwreck],
  X = labels_x[!isShipwreck],
```

```
  Point = labels_point[!isShipwreck]
)
```

Create a list aiming to assign shipwreck observations to workshop productions:

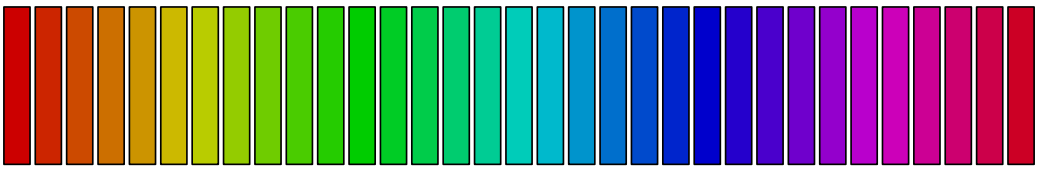
```
labels_list_Shipwreck <- list(
  Code = labels_code[!isTrueIND],
  Cross = labels_cross[!isTrueIND],
  X = labels_x[!isTrueIND],
  Point = labels_point[!isTrueIND]
)
```

Build two other lists containing named group color vectors, picking different colors within the **rainbow** palette:

```
color_list <- list()

for (i in 1:length(factor_list)){
  cv <- rainbow(nlevels(factor_list[[i]]), v=.8)
  color_list[[i]] <- cv
  names(color_list)[i] = names(factor_list)[i]
}

color_list_Shipwreck <- list()

for (i in 1:length(factor_list_Shipwreck)){
  cv <- rainbow(nlevels(factor_list_Shipwreck[[i]]), v=.8)
  color_list_Shipwreck[[i]] <- cv
  names(color_list_Shipwreck)[i] = names(factor_list_Shipwreck)[i]
}
```

To visualize the colors:

```
par(mar = c(0,0,0,0))
barplot(rep(100, nlevels(factor_list$ProvGroup)),
        col = color_list$ProvGroup,
        axes = FALSE)
```



2.8 Enunciate exception columns

Create a vector that enunciate which ordinal variables have "none" as a exceptional value when calculating the distance between values.

```
excep_cols <- c("INCLUS_DISTRIB","INCLUS_ORIENT","COAR_ROUNDNESS",
                "COAR_FORM","COAR_SPACING","COAR_SORTING","FINE_FORM")
```

In order to understand this "exceptional value" feature, compare the levels of regular and exceptional variables. However, to do that at this point you must re-assure the order of petrographic variables (i.e. format factors levels):

```
cleanAmphorae <- order_petro(cleanAmphorae)
```

This step is not necessary for applying the protocols because the apply_ordination function already does it internally, before calculating distances.

Variable	Values
INCLUS_DISTRIB	poorly, poorly to moderately, moderately, moderately to well, well, none
TEMP	unfired, 700-800oC, 800-900oC, 900-1000oC, 1000-1100oC
COAR_FREQ	none, very few, few, common, abundant, very abundant
COAR_ROUNDNESS	angular, angular to subangular, subangular, subangular to subrounded, subrounded, subrounded to rounded, rounded, none
COAR_R_CALS	none, few, common, frequent, dominant, predominant
FINE_FORM	elongate, elongate to equidimensional, equidimensional, equidimensional to laminar, laminar, none
FINE_C_QTZ	none, few, frequent, predominant

2.9 Choose geochemical data

```
chemVars16 <- c("Fe2O3","Al2O3","TiO2","MgO","CaO","SiO2",
                "Th","Nb","Zr","Y","Ce","Ga","V","Zn","Ni","Cr")
```

2.10 Save transformed geochemical data to file (optional)

There is no need to save it in the environment, because **apply_ordination** will transform the data internally and save the results in "ordination_object$transformed_data", when applicable.

```
write(transform_coda(cleanAmphorae,
                     coda_variables = chemVars16,
                     method = c("CLR")),
      file = paste(directories$transCoDa,
                   "transAmphorae_clr.csv",
                   sep = "/"))
```

In the output table, columns will be ordered as:

1. variables not transformed,
2. Raw version of the selected variables,
3. Transformed version of the selected variables.

2.11 Other CoDa packages

Be aware that compositional data (CoDa) analysis can be much more complex than what cerUB currently allows for. For more possibilities, you may explore other R packages: *compositions*, *zCompositions*, and *robCompositions*.

However, before jumping into using more complex techniques, we do recommend a deeper introduction to CoDa:

> Pawlowsky-Glahn, V., Buccianti, A., 2011. Compositional data analysis: theory and applications. Wiley.

< back to index

3 Protocol 1 - Geochemical data

The following example applies protocol 1 to confirm the workshops' chemical reference groups.

Protocol 1 consist in:

1. Select ***geochemical*** compositional data (CoDa);
2. Apply ***transformation***;
3. Perform ***robust Principal Components Analysis*** (robPCA), implicitly using Euclidean distance;
4. Perform ***PERMANOVA & PERMDISP*** tests;

Last, search for outliers and re-do protocol excluding outliers.

NOTE: The initial procedures must be ran at least once before any protocol can be applied.

3.1 Ordination procedure

```
prot1 <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                          "1", # select protocol 1
                          coda_override = chemVars16,
                          coda_transformation = "ILR")
#> sROC 0.1-2 loaded
#> [1] "78.24% of variance explained in 2D"
#> [1] "86.54% of variance explained in 3D"
#> [1] "Protocol 1 ended."
```

The outcome is an *ordination object*. In this case, it is the output of **pcaCoDa** function in *robCompositions* package, in addition to several extra information, such as the transformed data ('transformed_data'), the distance matrix ('dist_matrix'), and the ready-to-plot texts indicating the fitness of the 2D/3D projections respect the distance matrix ('sub2d', 'sub3d'). The later are printed in the console once the object is created.

```
class(prot1)
#> [1] "pcaCoDa"

names(prot1)
#>  [1] "scores"                "loadings"
#>  [3] "eigenvalues"           "method"
#>  [5] "princompOutputClr"     "mult_comp"
#>  [7] "seed"                  "init_seed"
#>  [9] "samples"               "sub2D"
```

```
#> [11] "sub3D"                  "transformation_method"
#> [13] "transformed_data"       "dist_matrix"
#> [15] "name"
```

3.2 Simplify CoDa names

We may want to simplify the names of the transformed variables before plotting them in a biplot. The **transform_coda** function, which is called inside **apply_ordination** for protocol 1, generates composite names with format "transformationMethod-component" for all transformed variables (e.g., "CLR-Fe2O3"). The **simplify_coda_names** function replaces these names back to the shorter version (e.g., "Fe2O3"). However, you must always remember that the variables projected in biplots are not the originals but the transformed versions. This is particularly important when dealing with log-ratio variables since they contain information that goes beyond the original variable (i.e., divider).

```
prot1 <- simplify_coda_names(prot1)
```

3.3 Test the given chemical reference groups

Perform four tests (**anosim**, **betadisper**, **permdisp2**, and **permanova**) that assess the separation and uniformity of the given group factor. For more details on these tests, we refer to:

> Anderson, M.J., Walsh, D.C.I., 2013. PERMANOVA, ANOSIM, and the Mantel test in the face of heterogeneous dispersions: What null hypothesis are you testing? Ecol. Monogr. 83, 557-574. doi:10.1890/12-2010.1

The whole test batch may take several minutes depending on the size of the data matrix and the number of groups.

```
prot1_tests <- test_groups(prot1$dist_matrix, factor_list$ChemGroup)
#> [1] "initiating test batch..."
#> [1] "vegan::anosim done."
#> [1] "vegan::betadisper done."
#> [1] "vegan::permutest done."
#> [1] "vegan::adonis done."
#> [1] "Test batch completed."
```

The tests outputs can be accessed by their name:

```
names(prot1_tests)
#> [1] "permanova" "betadisp"  "permdisp2" "anosim"    "text"
```

The object also contains a "text" object, which is a function that generates a list of text lines for plotting the results of PERMANOVA and PERMDISP2 tests. It may feel confusing, but keep in mind that this "portable" function requires the same ordination object as an argument.

```
displayTestText <- function(test_text) {
  par(mar = c(0, 0, 0, 0), fig = c(0.05, 0.9, 0.05, 0.9))
  plot.new()
  for (i in 1:length(test_text)) {

    # this is for calculating the vertical
    # position of paragraphs and lines
    test_spacing_paragraph = 0.8
    test_spacing_line = 0.8

    first_line_pos_y <-
      1 - test_spacing_paragraph * ( (i - .9) / length(test_text) )

    pos_y <- first_line_pos_y

    if (length(test_text[[i]]) > 1) {

      next_paragraph_pos_y <-
        1 - test_spacing_paragraph * ( i / length(test_text) )

      for (j in 2:length((test_text[[i]])))
      {
        pos_y <-
          c(pos_y,
            first_line_pos_y - test_spacing_line *
              ((j - 1) / (length((test_text[[i]])))) *
              (first_line_pos_y - next_paragraph_pos_y)
          )
      }
    }
    # display a paragraph (a element of the list)
    text(0, pos_y, labels = test_text[[i]], cex = 0.8, pos = 4)
  }
}
```

```
displayTestText(prot1_tests$text(prot1_tests))
```

PERMANOVA:
$F = 30.845$ ($p \leq 0.001$)
$R^2 = 0.929$

PERMDISP2:
$F = 1.95$ ($p = 0.002$)

A rule-of-thumb for interpreting PERMANOVA and PERMDISP2 results is: if both p-values are low enough (e.g. < 0.05), the classification given is a good approximation of the data.

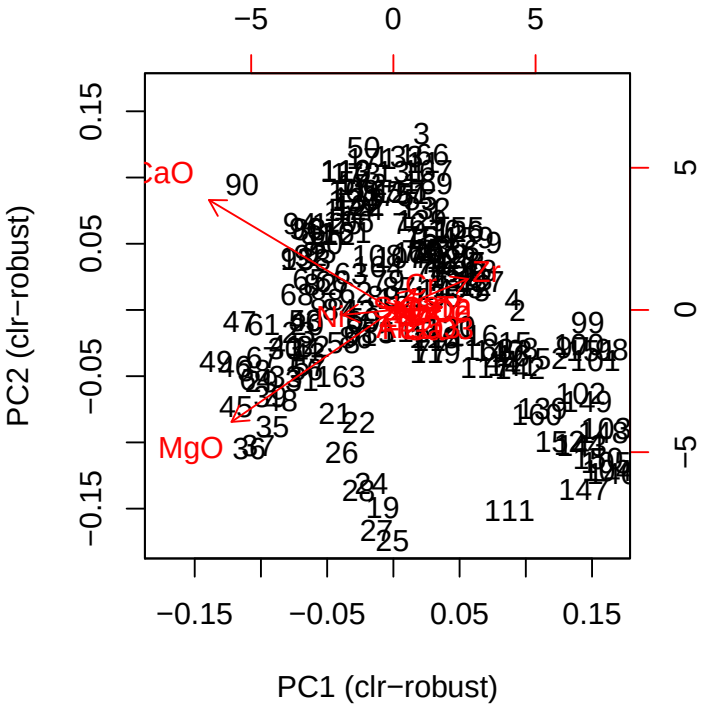


Figure 1: Default biplot in R

3.4 Biplots

Ordination objects are best represented in biplots, which simultaneously display the projections of observations (points) and variables (arrows) over the same space. There are several options for creating biplots in R, starting with the readily available **biplot** function:

```
biplot(prot1)
```

Although there are several options for customizing this default biplot function, we recommend the use of the *biplot2d3d* package. This package wraps a lot of possibilities in R.

```
library(biplot2d3d)
```

The *biplot2d3d* package use functions of other packages to allow the customization of virtually all aspects of a biplot. Another important feature of this package is the creation of three-dimensional interactive biplots using the *rgl* package.

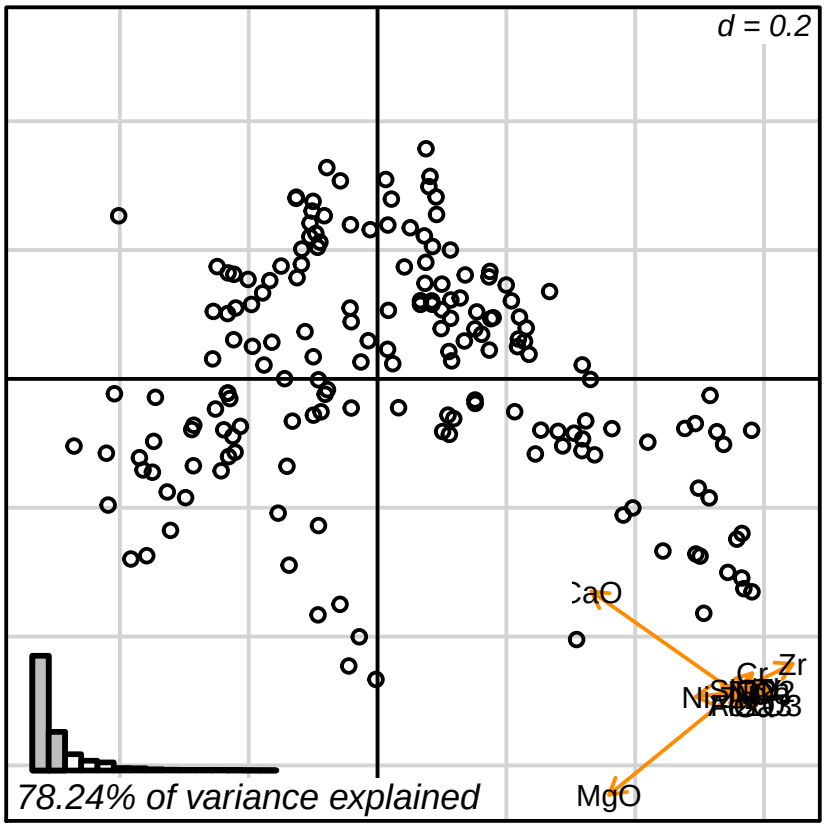


Figure 2: default 2D

3.4.1 Biplot 2D

You can consult all tuning options available in the **biplot_2d** function by calling up the help file:

```
?biplot_2d
```

The default configuration will probably give you a much clearer picture than the **biplot** function, specially if your dataset contains more than 100 observations.

```
biplot_2d(prot1)
```

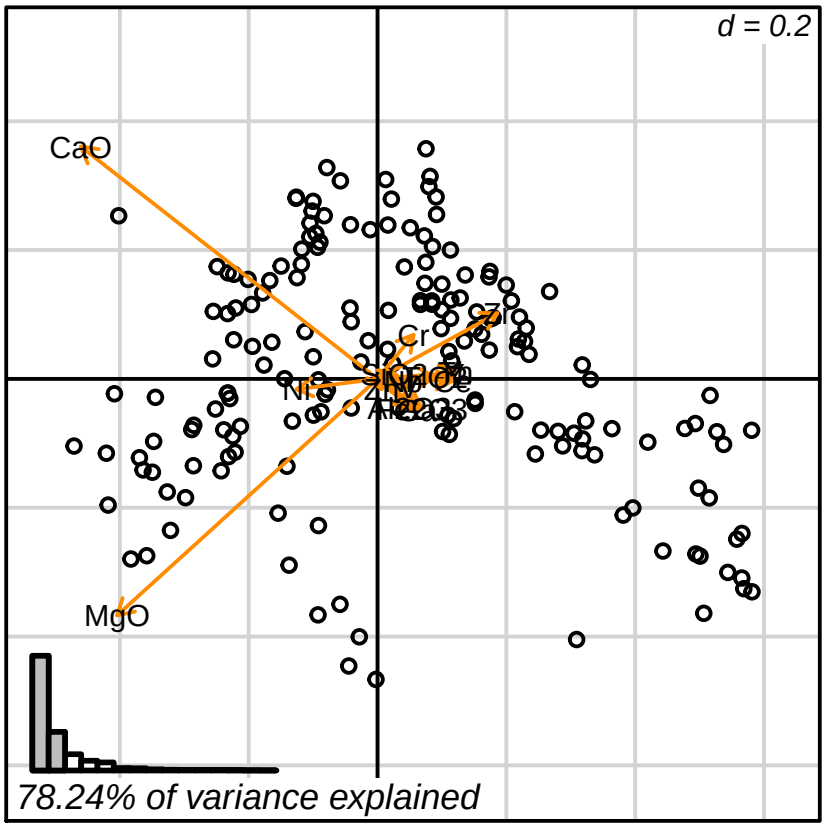


Figure 3: detach_arrows = FALSE

The default setting detaches the variables projections (arrows) and places them as a miniature in the bottom-right corner. In this format they may still be interpreted, much like the North when reading a map. Remember though: the more longer arrows you see, the less each one of them is reliable when comparing point values. Here, we were 'lucky' for getting two nearly orthogonal variables (CaO and MgO), which means, for instance, that those observations in the top-left corner are surely more calcareous than those in the bottom-right. See the Appendix section for more details.

If detaching the arrows is not of your preference, you can disable this:

```
biplot_2d(prot1,
          detach_arrows = FALSE,
          output_type = "preview")
```

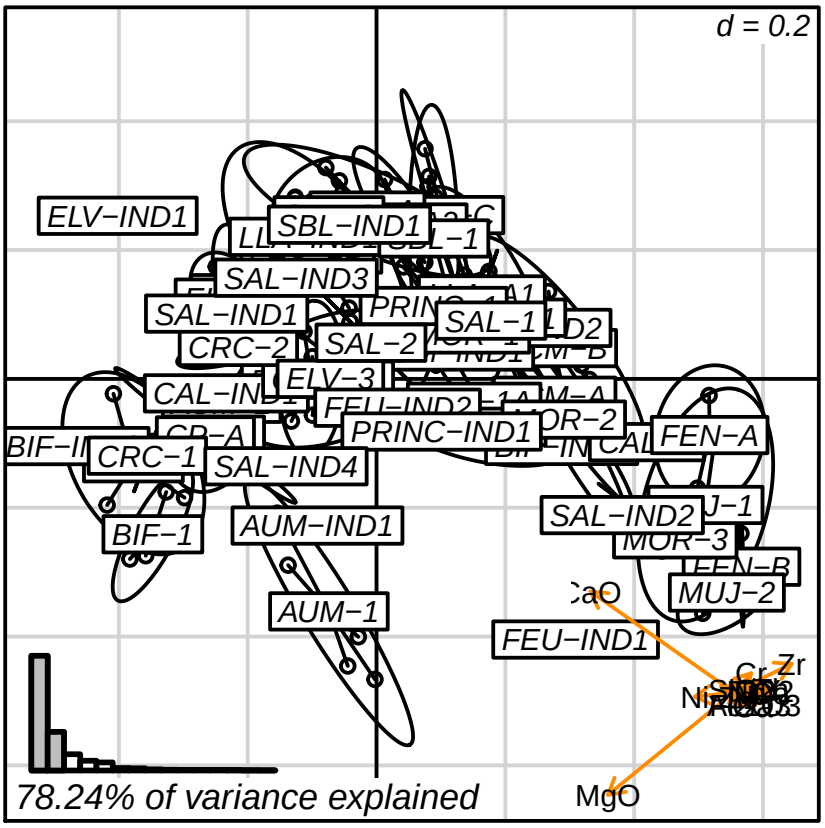


Figure 4: default with groups

You can also visualize how the projection of points respond to a given typology (in this case, the chemical reference groups defined in previous studies).

```
biplot_2d(prot1,
          groups = factor_list$ChemGroup,
          output_type = "preview")
```

To get a prettier plot or match your research needs, you can play with the options given by the **biplot_2d**.
Note that groups are by default marked using inertia ellipses. They can only be interpreted as confidence ellipses if each group can be assumed to be normally distributed in all variables considered (see more details on the scaling factor "group_ellipse_cex" in the help file). This is often reasonable concerning groups that are either too small (< 10) or contain subgroups.
In the argument "test_text" you can introduce the "text" function of the "prot1_tests" object.

```
biplot_2d(prot1,
          groups = factor_list$ChemGroup,
          group_color = color_list$ChemGroup,
          group_label_cex = 0.6,
          invert_coordinates = c(TRUE, TRUE),
          arrow_label_cex = 0.7,
          test_text = prot1_tests$text(prot1_tests),
          test_cex = 0.8,
          test_fig = c(0, 0.5, 0.65, .99),
          output_type = "preview")
```

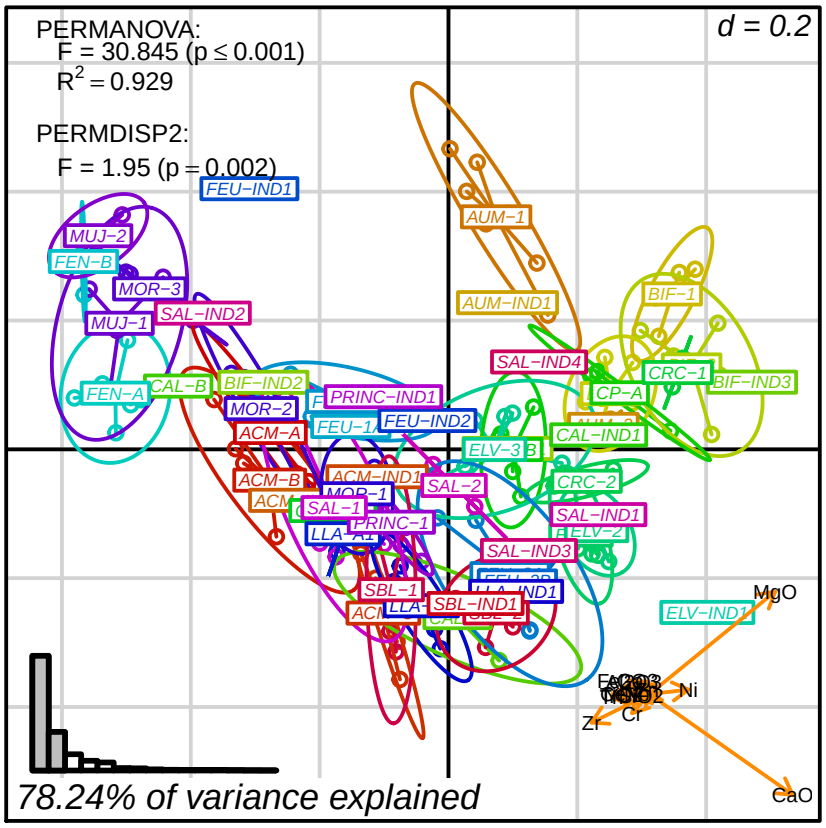


Figure 5: tunning appearance, groups with colors

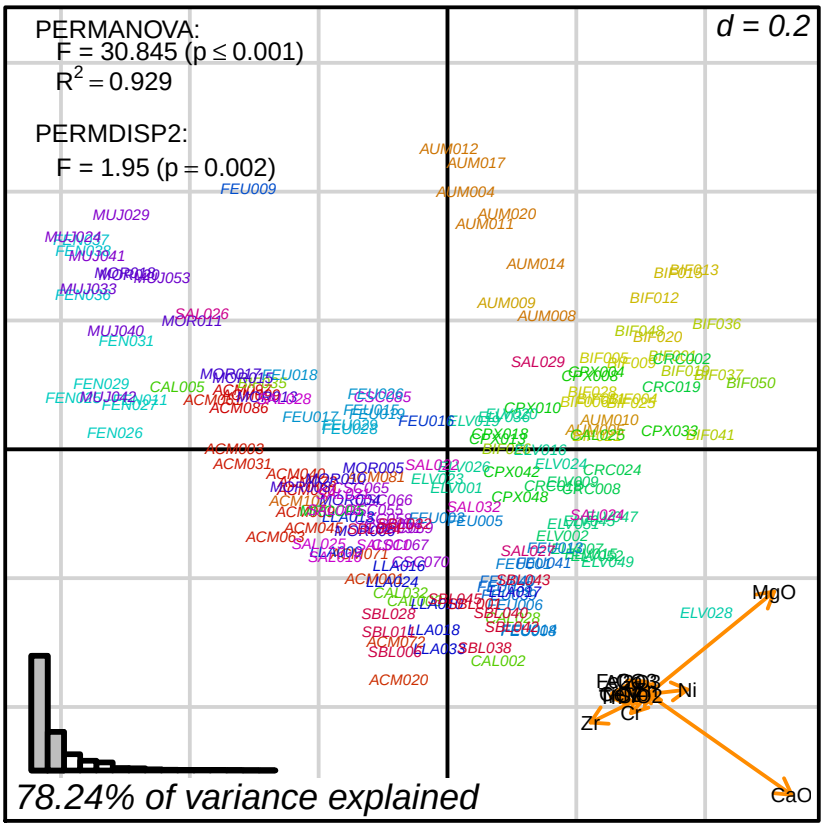


Figure 6: labeled points

```
biplot_2d(prot1,
          groups = factor_list$ChemGroup,
          group_color = color_list$ChemGroup,
          group_label_cex = 0,
          group_star_cex = 0,
          group_ellipse_cex = 0,
          point_type = "label",
          point_label = row.names(cleanAmphorae)[!isShipwreck],
          point_label_cex = 0.5,
          invert_coordinates = c(TRUE, TRUE),
          arrow_label_cex = 0.7,
          test_text = prot1_tests$text(prot1_tests),
          test_cex = 0.8,
          test_fig = c(0, 0.5, 0.65, .99),
          output_type = "preview")
```

It is also possible to save 2D biplots into various file formats (png, tiff, jpeg, eps):

```
# better PNG version
biplot_2d(prot1,
          ordination_method = "PCA",
          invert_coordinates = c (TRUE,TRUE),
          grid_cex = 2.5,
          ylim = c(-.1,.1),
          point_type = "point",
          groups = factor_list$ChemGroup,
          group_color = color_list$ChemGroup,
          group_label_cex = 1.5,
          arrow_label_cex = 2,
          arrow_cex = 0.2,
          arrow_lwd = 2.5,
          arrow_fig = c(.6,.95,0,.35),
          subtitle_cex = 2.5,
          test_text = prot1_tests$text(prot1_tests),
          test_fig =c(0, 0.5, 0.62, .99),
          test_cex = 2,
          fitAnalysis_fig = c(0,.7,.05,.5),
          # saving settings
          file_name = "Prot1_Biplot2D",
          directory = directories$prot1,
          width = 1000, height = 1000,
          output_type = "png")
```

3.4.2 Biplot 3D

Most 2D options are also available when generating 3D biplots. Consult the help file for details.

```
?biplot_3d
```

Unfortunately, we are not able to reproduce all features of a rgl device (interactivity) in this document:

```
biplot_3d(prot1,
          ordination_method = "PCA",
          groups = factor_list$ChemGroup,
          group_color = color_list$ChemGroup,
          point_type = "point",
          group_representation = "stars",
          star_centroid_radius = 0,
          star_label_cex = .8,
          test_text = prot1_tests$text(prot1_tests),
          test_cex = 1.25,
          test_fig = c(0, 0.5, 0.65, .99))
```

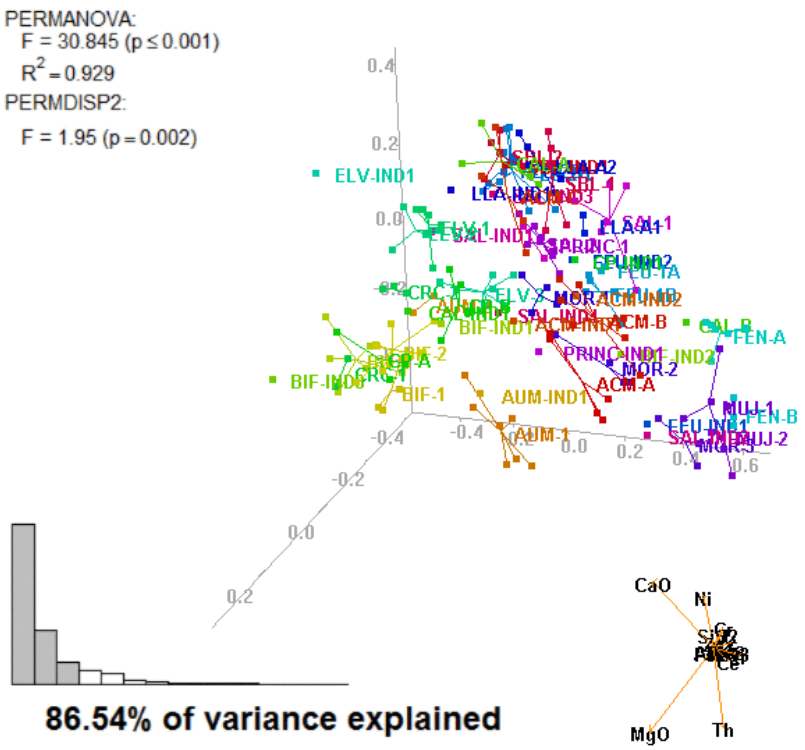


Figure 7: Prot1_Biplot3D_snapshot.png

You can save an animated GIF and a PNG snapshot using the **animation** function.

```
biplot2d3d::animation(directory = directories$prot1,
                      file_name = "Prot1_Biplot3D")
```

You will need to install ImageMagick to be able to generate the GIF animation.

In these images, you get what you would see when running the biplot_3d function in a regular R session.

NOTE: Animated GIF will not be displayed in the pdf version of this document.

3.5 Comparing CoDa transformations

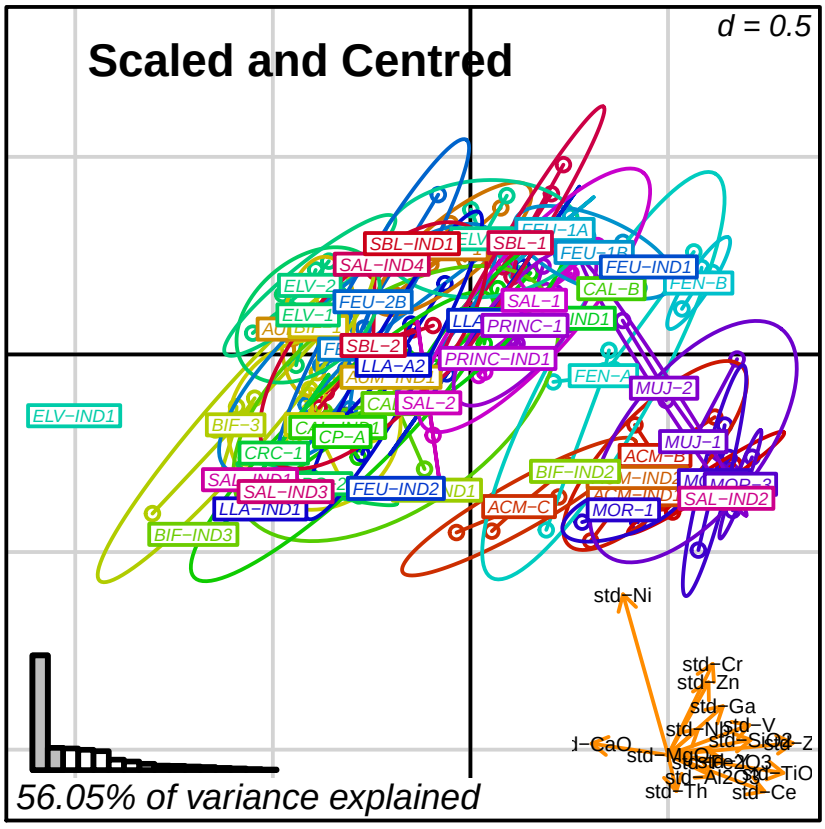
There is a lot of debate on which transformation is useful–or even *valid*–for analyzing geochemical compositions in Archaeometry. We show here how you can compare the results of applying different transformations to the same dataset.

First, create different ordination objects for each type of CoDa transformation that you wish to compare:

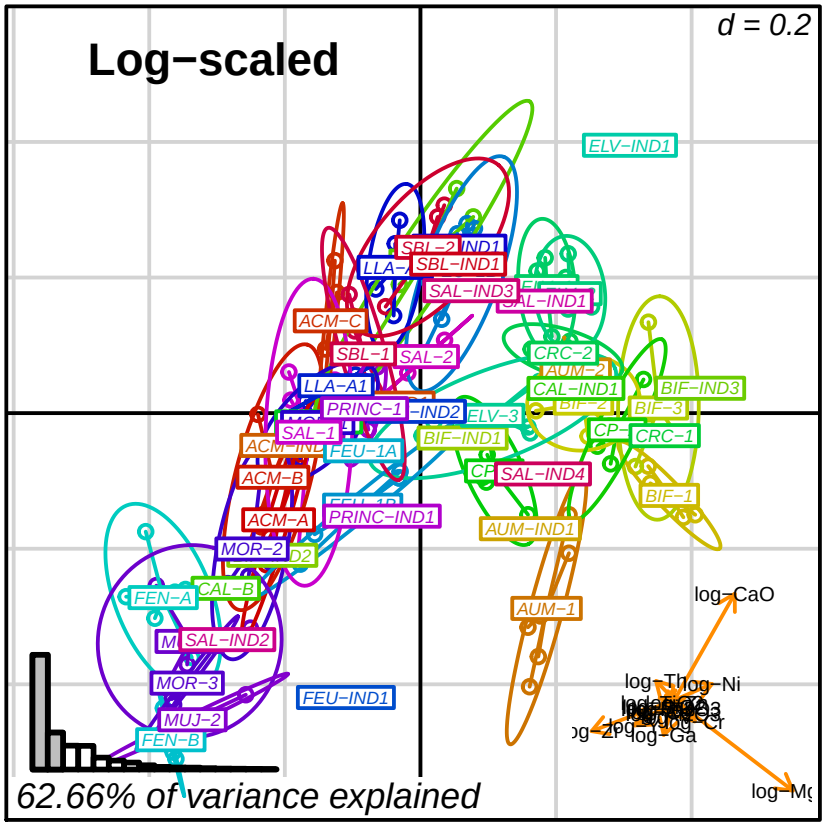
```
prot1_std <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "1", # select protocol 1
                              coda_override = chemVars16,
                              coda_transformation = "std")
#> [1] "56.05% of variance explained in 2D"
#> [1] "64.91% of variance explained in 3D"
#> [1] "Protocol 1 ended."

prot1_log <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "1", # select protocol 1
                              coda_override = chemVars16,
                              coda_transformation = "log")
#> [1] "62.66% of variance explained in 2D"
#> [1] "72.53% of variance explained in 3D"
#> [1] "Protocol 1 ended."

prot1_ALR <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "1", # select protocol 1
                              coda_override = chemVars16,
                              coda_transformation = "ALR",
                              # this is the divisor component
                              coda_alr_base = "Fe2O3")
#> [1] "70.3% of variance explained in 2D"
#> [1] "81.75% of variance explained in 3D"
#> [1] "Protocol 1 ended."

prot1_CLR <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "1", # select protocol 1
                              coda_override = chemVars16,
                              coda_transformation = "CLR")
#> [1] "64.16% of variance explained in 2D"
#> [1] "75.57% of variance explained in 3D"
#> [1] "Protocol 1 ended."
```

Then, create the respective biplots:

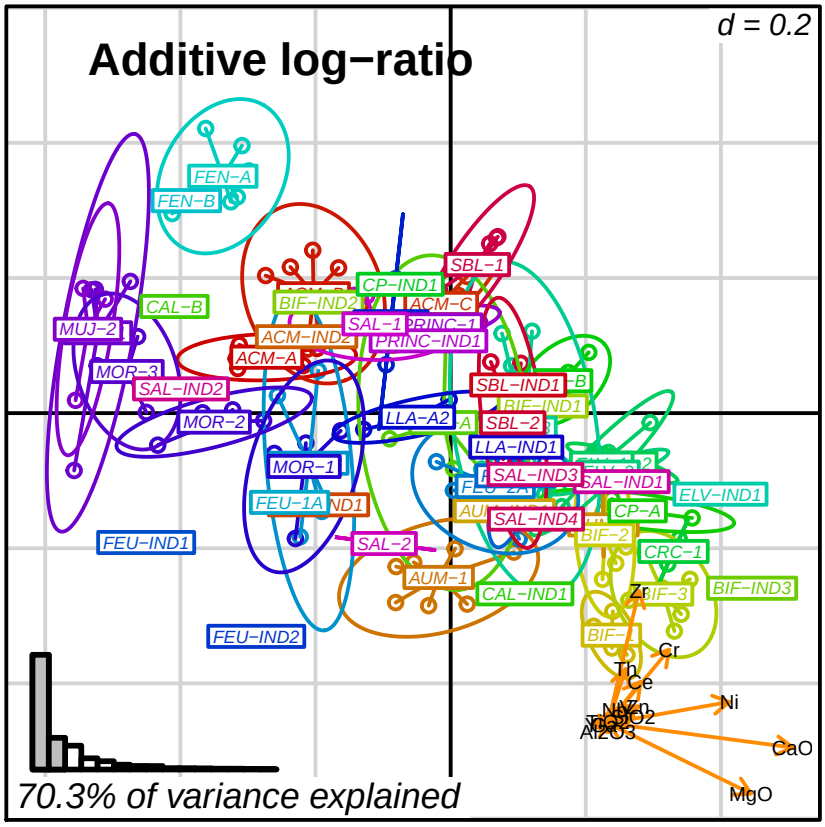
```
biplot2d3d::biplot_2d(prot1_std,
                       groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      group_label_cex = 0.6,
                      arrow_label_cex = 0.7,
                      ylim = c(-0.9, 0.6),
                      x_title = "Scaled and Centred",
                      x_title_cex = 1.5,
                      x_title_fig = c(0.05, 0.9, 0.85, 1),
                      output_type = "preview")
```



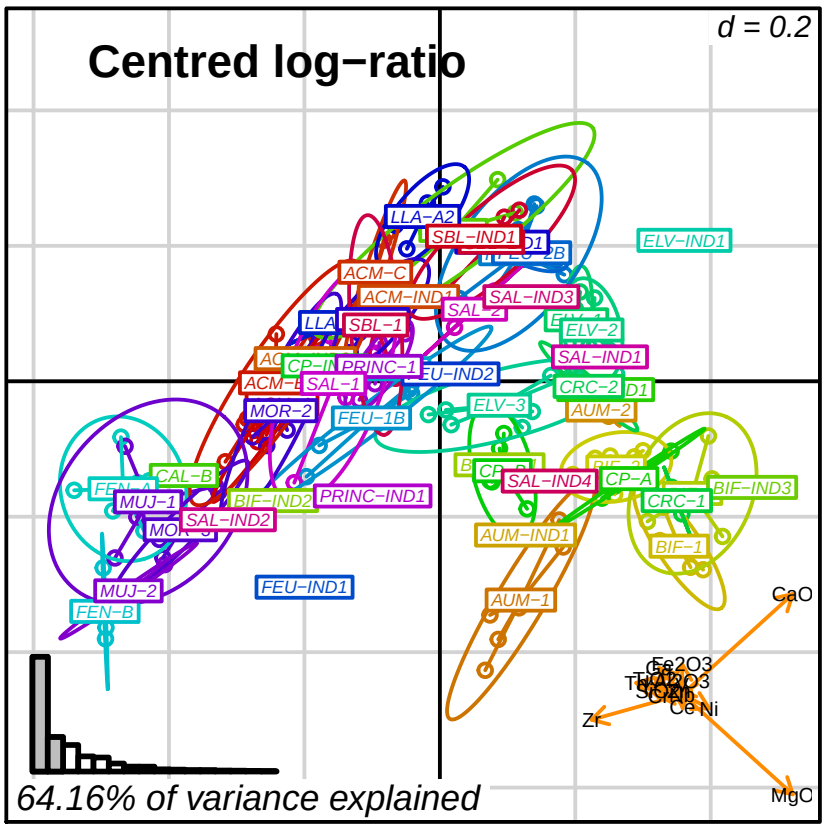
```
biplot2d3d::biplot_2d(prot1_log,
                       groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      group_label_cex = 0.6,
                      arrow_label_cex = 0.7,
                      ylim = c(-0.6, 0.6),
                      x_title = "Log-scaled",
                      x_title_cex = 1.5,
                      x_title_fig = c(0.05, 0.9, 0.85, 1),
                      output_type = "preview")
```



```
biplot2d3d::biplot_2d(prot1_ALR,
                       groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      group_label_cex = 0.6,
                      arrow_label_cex = 0.7,
                      ylim = c(-0.6, 0.6),
                      x_title = "Additive log-ratio",
                      x_title_cex = 1.5,
                      x_title_fig = c(0.05, 0.9, 0.85, 1),
                      output_type = "preview")
```



```
biplot2d3d::biplot_2d(prot1_CLR,
                       groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      group_label_cex = 0.6,
                      arrow_label_cex = 0.7,
                      ylim = c(-0.5, 0.4),
                      x_title = "Centred log-ratio",
                      x_title_cex = 1.5,
                      x_title_fig = c(0.05, 0.9, 0.85, 1),
                      output_type = "preview")
```



< back to index

4 Protocol 2 - Petrographic data

The following example applies protocol 2 to confirm workshops' petrographic groups.

Protocol 2 consist in:

1. Select ordinal ***petrographic*** data;
2. Transform to ***ranks***;
3. ***Extended Gower distance***, using:

a. **Relative ranking difference** (RRD)
b. **Neighbor interchange** (NI)

4. Apply ordination procedure:

a. ***Principal Coordinates Analysis*** (PCoA)
b. ***Non-metric Dimensional Scaling*** (NMDS)

5. Perform ***PERMANOVA & PERMDISP*** tests;

Last, search for outliers and re-do protocol excluding outliers.

NOTE: The initial procedures must be ran at least once before any protocol can be applied.

The key references on the Extended Gower distance are:

Pavoine, S., Vallet, J., Dufour, A.-B., Gachet, S., Daniel, H., 2009. On the challenge of treating various types of variables: application for improving the measurement of functional diversity. Oikos 118, 391-402. doi:10.1111/j.1600-0706.2008.16668.x

Podani, J., 1999. Extending Gower's General Coefficient of Similarity to Ordinal Characters on JSTOR. Taxon 48, 331-340. doi:10.2307/1224438

Gower, J.C., 1971. A General Coefficient of Similarity and Some of Its Properties. Biometrics 27, 857-871. doi:10.2307/2528823

4.1 Ordination procedure

Depending on which type of distance calculation (RRD/NI), protocol 2 performs different ordination methods (PCoA/NMDS). Both PCoA and NMDS require specifying the number of dimensions in which to project the data. Therefore, you must generate specific 2D and 3D ordination objects:

```
prot2a_2d <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "2a", # select protocol 2a (RRD & PCoA)
                              exception_columns = excep_cols,
                              variable_tags = varCode)

prot2b_2d <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "2b", # select protocol 2a (NI & NMDS)
                              exception_columns = excep_cols,
                              variable_tags = varCode)

prot2a_3d <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "2a", # select protocol 2a (RRD & PCoA)
                              exception_columns = excep_cols,
                              variable_tags = varCode,
                              dimensions = 3)

prot2b_3d <- apply_ordination(cleanAmphorae[!isShipwreck,],
                              "2b", # select protocol 2a (NI & NMDS)
                              exception_columns = excep_cols,
                              variable_tags = varCode,
                              dimensions = 3)
```

The ordination objects generated with protocol 2 are different from those in protocol 1 since it uses different functions. However, the main components are still the same: the projection of observations or *scores* (**points**) and of variables or *loadings*.

```
class(prot2a_2d)
#> [1] "list"
names(prot2a_2d)
#>  [1] "points"        "eig"           "x"             "ac"
#>  [5] "GOF"           "sub2D"         "GOF2_2D"       "loadings"
#>  [9] "variable_tags" "name"          "dist_matrix"
class(prot2b_2d)
#> [1] "metaMDS" "monoMDS"
names(prot2b_2d)
#>  [1] "nobj"          "nfix"          "ndim"          "ndis"
#>  [5] "ngrp"          "diss"          "iidx"          "jidx"
#>  [9] "xinit"         "istart"        "isform"        "ities"
#> [13] "iregn"         "iscal"         "maxits"        "sratmx"
#> [17] "strmin"        "sfgrmn"        "dist"          "dhat"
#> [21] "points"        "stress"        "grstress"      "iters"
#> [25] "icause"        "call"          "model"         "distmethod"
#> [29] "distcall"      "distance"      "converged"     "tries"
#> [33] "engine"        "species"       "data"          "init_seed"
#> [37] "trymax"        "sub_stress"    "sub2D"         "GOF2_2D"
#> [41] "loadings"      "variable_tags" "name"          "dist_matrix"
```

4.2 Test the given fabric groups

The fabric groups defined in previous studies can be tested against Protocol 2 distance matrices. We can use either "prot2a_2d$dist_matrix" or "prot2a_3d$dist_matrix", because they are the same. Remember that this test batch may take several minutes.

```
prot2a_tests <- test_groups(prot2a_2d$dist_matrix,
                            factor_list$FabricGroup)
prot2b_tests <- test_groups(prot2b_2d$dist_matrix,
                            factor_list$FabricGroup)
```

These tests were explained in protocol 1.

4.3 Biplots

The details on how to create biplots is already explained in protocol 1. Concerning protocol 2, we can compare the results of version *2a* (RRD, PCoA) and *2b* (NI, NMDS).

4.3.1 Biplot 2D

```
arrows_label_adj <- rbind(c(.5,.8),c(.5,1),c(.5,1),c(.5,0),c(.5,1),
                          c(.5,0),c(0,.5))
row.names(arrows_label_adj) <- c("L48","L24","L5","L36","S7",
                                 "S8","S11")

biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = 0.5,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot2a_2d$sub2D,
                      test_text = prot2a_tests$text(prot2a_tests),
                      test_cex = 0.8,
```

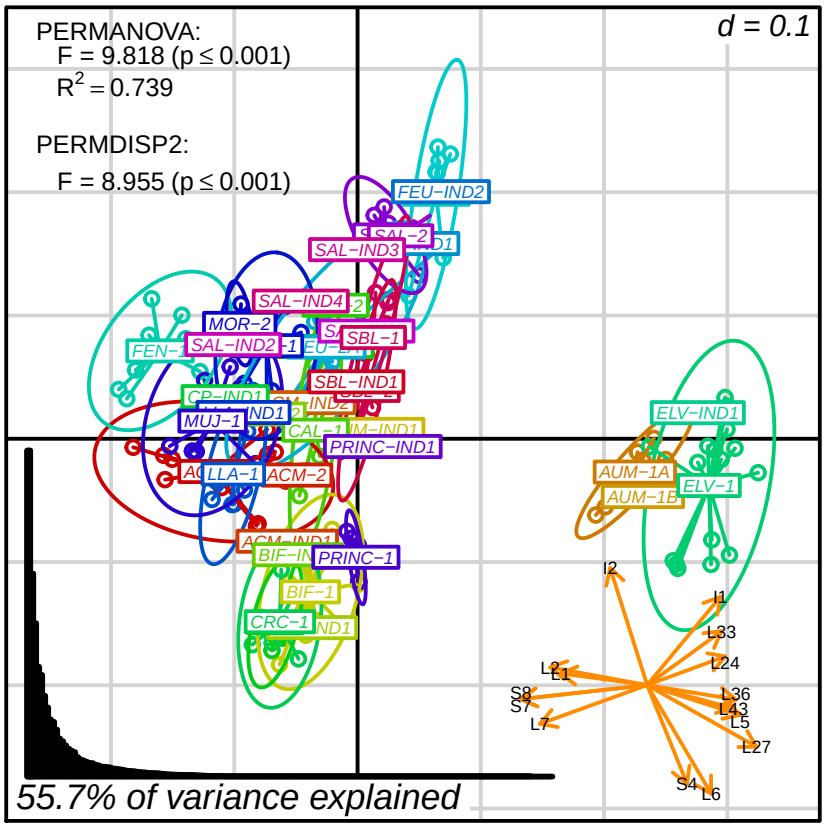


Figure 8: protocol 2a

```
                    test_fig = c(0, 0.5, 0.62, .99),
                    fitAnalysis_fig = c(0,.7,.05,.5),
                    output_type = "preview")
```

```
arrows_label_adj <- rbind(c(.5,1),c(.5,0),c(.5,1),c(.5,1),c(.5,0),
                          c(0,.5),c(1,.5))
row.names(arrows_label_adj) <- c("S7","S8","CLAY","L24","L43",
                                 "L5","L36")

biplot2d3d::biplot_2d(prot2b_2d,
                      ordination_method = "NMDS",
                      xlim = c(-.42,.38),
                      ylim = c(-.45,.25),
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot2b_2d$sub2D,
                      test_text = prot2b_tests$text(prot2b_tests),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.62, .99),
                      fitAnalysis_stress_axis_cex = 0.8,
                      fitAnalysis_fig = c(.1,.6,.1,.4),
                      output_type = "preview")
```

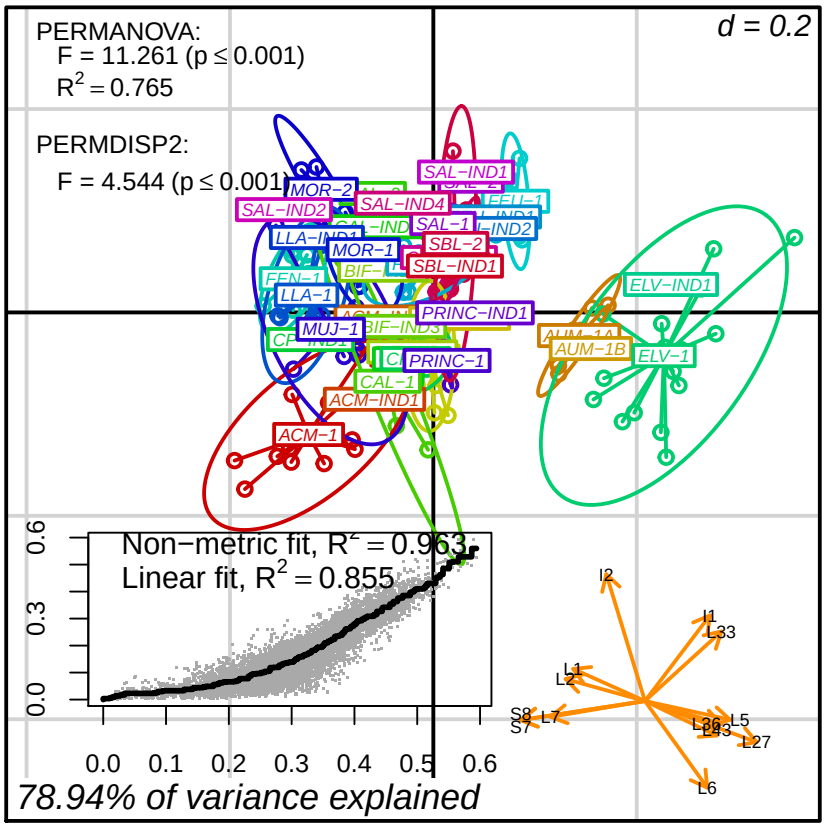


Figure 9: protocol 2b

4.3.2 Biplot 3D

```
biplot2d3d::biplot_3d(prot2a_3d,
                      ordination_method = "PCoA",
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot2a_3d$sub3D,
                      test_text = prot2a_tests$text(prot2a_tests),
                      test_cex = 1.25,
                      test_fig = c(0, 0.5, 0.65, .99),
                      view_zoom = 0.9)

biplot2d3d::animation(directory = directories$prot2,
                      file_name = "Prot2a_Biplot3D")
```

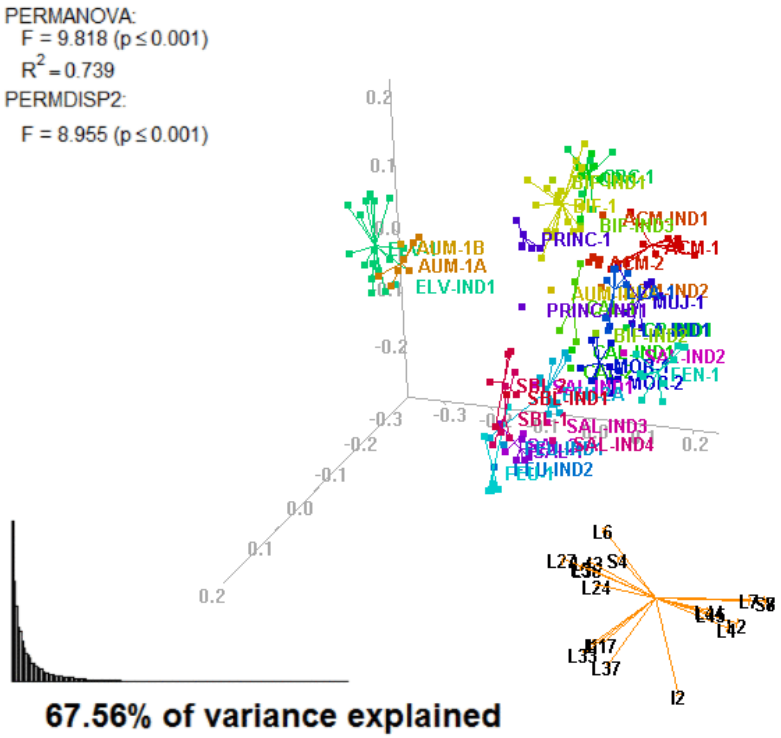


Figure 10: Prot2a_Biplot3D_snapshot.png

```
biplot2d3d::biplot_3d(prot2b_3d,
                      ordination_method = "NMDS",
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot2b_3d$sub3D,
                      test_text = prot2b_tests$text(prot2b_tests),
                      test_cex = 1.25,
                      test_fig = c(0, 0.5, 0.65, .99),
                      view_zoom = 0.9)

biplot2d3d::animation(directory = directories$prot2,
                      file_name = "Prot2b_Biplot3D")
```

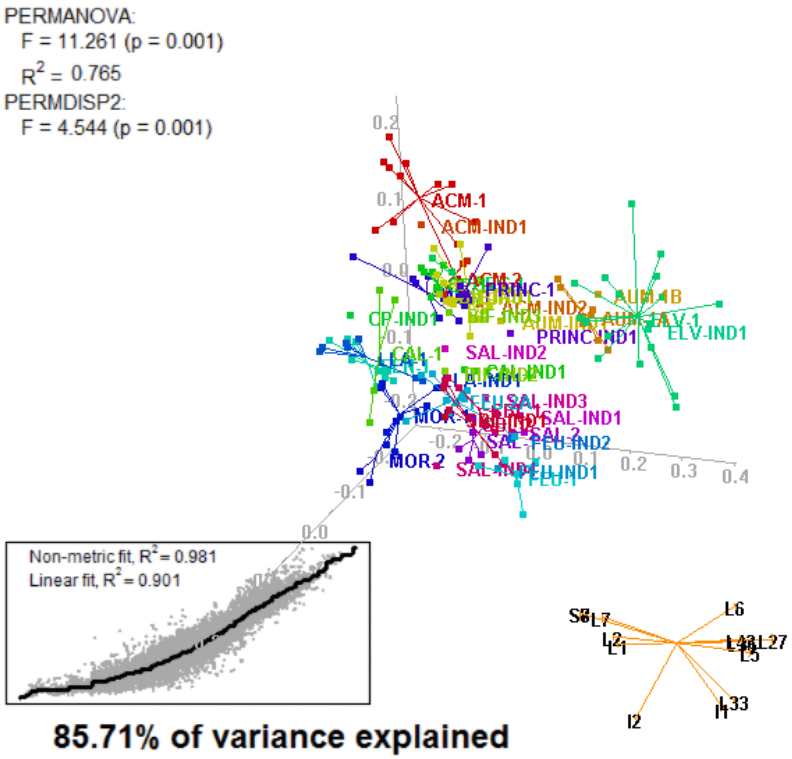


Figure 11: Prot2b_Biplot3D_snapshot.png

< back to index

5 Protocol 3 - Geochemical and petrographic data

The following example applies protocol 3 to confirm workshops' provenance groups.

Protocol 3 consist in:

1. Select ***geochemical*** compositional data (CoDa) and ordinal ***petrographic*** data;
2. ***Centred log-ratio transformation*** (clr) and transform to ***ranks***;
3. ***Extended Gower distance***, using **Relative ranking difference** (RRD);
4. Apply ***Principal Coordinates Analysis*** (PCoA);
5. Perform ***PERMANOVA & PERMDISP*** tests;

Last, search for outliers and re-do protocol excluding outliers.

NOTE: The initial procedures must be ran at least once before any protocol can be applied.

See protocol 2, for consulting references on the extended Gower distance.

5.1 Ordination procedure

Protocol 3 performs PCoA on a distance matrix calculated with Extended Gower coefficient of dissimilarity, combining Euclidean distances on transformed compositional data (50%) and RRD on ranked petrographic data (50%). As in protocol 2, PCoA requires specifying the number of dimensions and so you must 2D and 3D ordination objects separately:

```
prot3_2d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                             "3", # select protocol 3
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR")

prot3_3d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                             "3", # select protocol 3
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR",
                             dimensions = 3)
```

5.2 Simplify CoDa names

We may want to simplify the names of the transformed variables before plotting them in a biplot.

```
prot3_2d <- simplify_coda_names(prot3_2d)
prot3_3d <- simplify_coda_names(prot3_3d)
```

5.3 Test the given provenance groups

Because protocol 3 uses both geochemical and petrographic information, we can test the provenance assigned to the amphorae samples.

```
prot3_tests <- test_groups(prot3_2d$dist_matrix,
                           factor_list$ProvGroup)
```

These tests were explained in protocol 1.

5.4 Biplots

The details on how to create biplots is already explained in protocol 1. Unlike protocol 2, protocol 3 only generates one kind of projection (RRD, PCoA).

5.4.1 Biplot 2D

```
arrows_label_adj <- rbind(c(.5,.8),c(.5,1),c(.5,1),c(.5,0),c(.5,1),
                          c(.5,0),c(0,.5))
row.names(arrows_label_adj) <- c("L48","L24","L5","L36","S7",
                                 "S8","S11")

biplot2d3d::biplot_2d(prot3_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      ylim = c(-.3,.29),
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = 0.5,
                      arrow_label_cex = 0.6,
```

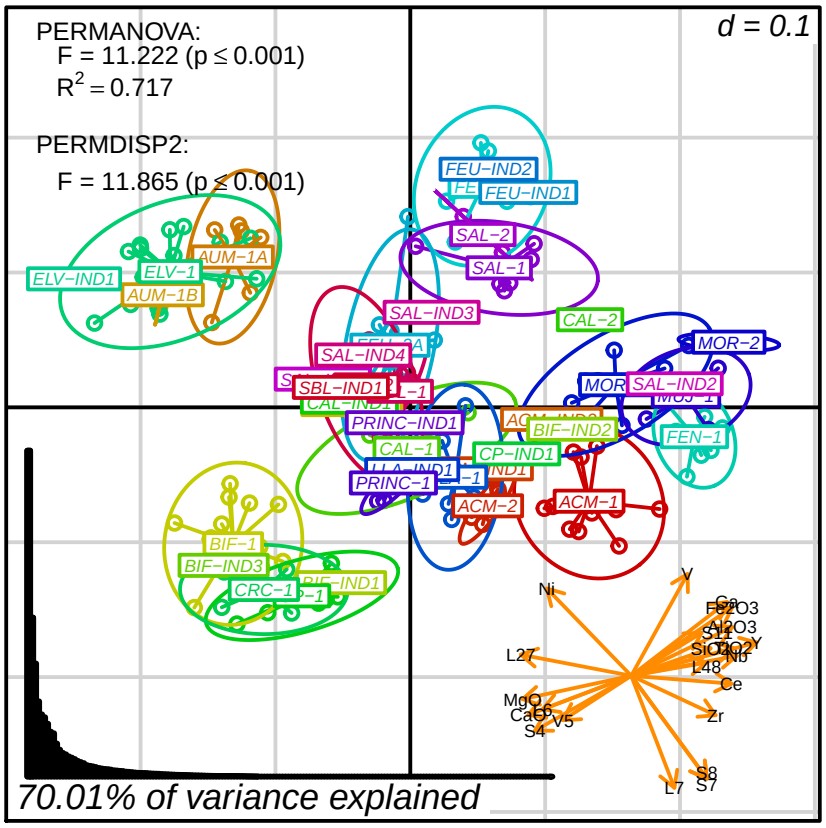


Figure 12: protocol 3

```
                arrow_fig = c(.6,.95,0,.35),
                arrow_label_adj_override = arrows_label_adj,
                subtitle = prot3_2d$sub2D,
                test_text = prot3_tests$text(prot3_tests),
                test_cex = 0.8,
                test_fig = c(0, 0.5, 0.62, .99),
                fitAnalysis_fig = c(0,.7,.05,.5),
                output_type = "preview")
```

5.4.2 Biplot 3D

```
biplot2d3d::biplot_3d(prot3_3d,
                      ordination_method = "PCoA",
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot3_3d$sub3D,
                      test_text = prot3_tests$text(prot3_tests),
                      test_cex = 1.25,
                      test_fig = c(0, 0.5, 0.65, .99),
                      view_zoom = 0.9)

biplot2d3d::animation(directory = directories$prot3,
                      file_name = "Prot3_Biplot3D")
```

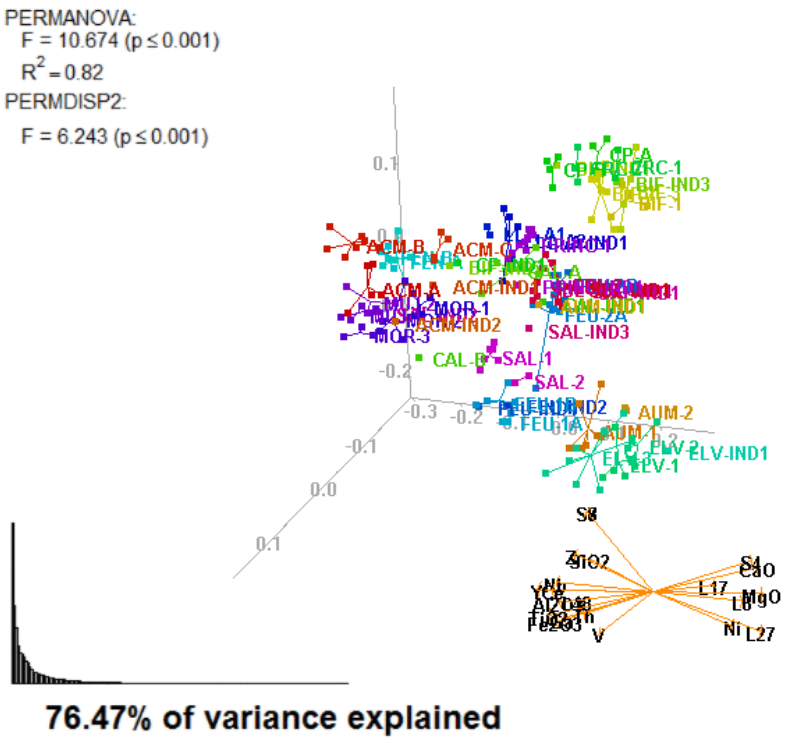


Figure 13: Prot3_Biplot3D.gif

< back to index

6 Protocol 4 - Provenance data

The following example applies protocol 4 to confirm workshops' provenance groups.

Protocol 4 consist in:

1. Select provenance-specific variables in ***geochemical*** compositional data (CoDa) and ordinal ***petrographic*** data;
2. ***Centred log-ratio transformation*** (clr) and transform to ***ranks***;
3. ***Extended Gower coefficient of dissimilarity***, using **Relative ranking difference** (RRD);
4. Apply ***Principal Coordinates Analysis*** (PCoA);
5. Perform ***PERMANOVA & PERMDISP*** tests;

Last, search for outliers and re-do protocol excluding outliers.

NOTE: The initial procedures must be ran at least once before any protocol can be applied.

6.1 Ordination procedure

As protocol 3, protocol 4 performs PCoA on a distance matrix calculated with Extended Gower coefficient of dissimilarity, combining Euclidean distances on transformed compositional data (50%) and RRD on ranked petrographic data (50%).

```
prot4_2d <- apply_ordination(cleanAmphorae[!isShipwreck,],
                             "4", # select protocol 4
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR")

prot4_3d <- apply_ordination(cleanAmphorae[!isShipwreck,],
                             "4", # select protocol 4
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR",
                             dimensions = 3)
```

However, protocol 4 uses a finer selection of petrographic variables, which are considered indicative of provenance (raw materials) rather than technology. Compare the number of variables in protocol 3 and 4:

Protocol 3	Protocol 4
78	59

6.2 Simplify CoDa names

We may want to simplify the names of the transformed variables before plotting them in a biplot.

```
prot4_2d <- simplify_coda_names(prot4_2d)
prot4_3d <- simplify_coda_names(prot4_3d)
```

6.3 Test the given provenance groups

With protocol 4, we can test the provenance assigned to the amphorae samples based only on provenance-specific variables.

```
prot4_tests <- test_groups(prot4_2d$dist_matrix,
                           factor_list$ProvGroup)
```

These tests were explained in protocol 1.

6.4 Biplots

The details on how to create biplots is already explained in protocol 1. As protocol 3, protocol 4 only generates one kind of projection (RRD, PCoA).

6.4.1 Biplot 2D

```
arrows_label_adj <- rbind(c(.5,1),c(0,0),c(1,.5),c(0,1),c(1,0),
                          c(0,.5),c(.5,1),c(1,.5),c(.5,1))
row.names(arrows_label_adj) <- c("CaO","S4","S7","S8","Ce",
                                 "Nb","Al2O3","S11","Fe2O3")

biplot2d3d::biplot_2d(prot4_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,FALSE),
                      ylim = c(-.35,.32),
                      point_type = "point",
```

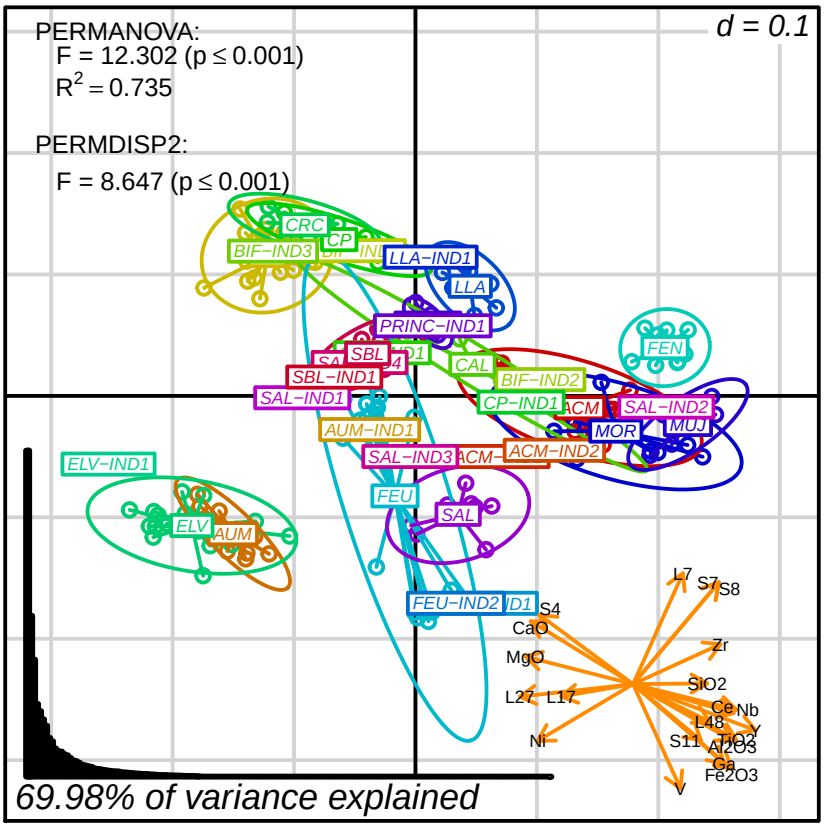


Figure 14: protocol 4

```
                groups = factor_list$ProvGroup,
                group_color = color_list$ProvGroup,
                group_label_cex = 0.6,
                arrow_mim_dist = .5,
                arrow_label_cex = 0.6,
                arrow_fig = c(.6,.95,0,.35),
                arrow_label_adj_override = arrows_label_adj,
                subtitle = prot4_2d$sub2D,
                test_text = prot4_tests$text(prot4_tests),
                test_cex = 0.8,
                test_fig = c(0, 0.5, 0.62, .99),
                fitAnalysis_fig = c(0,.7,.05,.5),
                output_type = "preview")
```

6.4.2 Biplot 3D

```
biplot2d3d::biplot_3d(prot4_3d,
                      ordination_method = "PCoA",
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot4_3d$sub3D,
                      test_text = prot4_tests$text(prot4_tests),
                      test_cex = 1.25,
                      test_fig = c(0, 0.5, 0.65, .99),
                      view_zoom = 0.9)

biplot2d3d::animation(directory = directories$prot4,
                       file_name = "Prot4_Biplot3D")
```

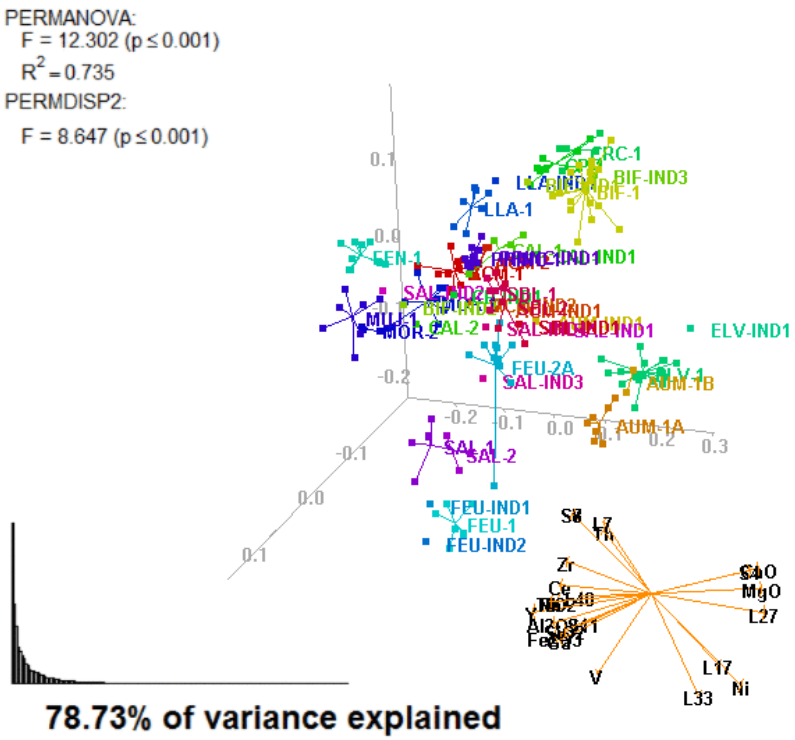


Figure 15: Prot4_Biplot3D.gif

< back to index

7 Protocol 4 - Provenance data with shipwrecks

The following example applies protocol 4 to confirm shipwrecks samples attribution to workshops' provenance groups.

Protocol 4 consist in:

1. Select provenance-specific variables in ***geochemical*** compositional data (CoDa) and ordinal ***petrographic*** data;
2. ***Centred log-ratio transformation*** (clr) and transform to ***ranks***;
3. ***Extended Gower coefficient of dissimilarity***, using **Relative ranking difference** (RRD);
4. Apply ***Principal Coordinates Analysis*** (PCoA);
5. Perform ***PERMANOVA & PERMDISP*** tests;

Last, search for outliers and re-do protocol excluding outliers.

NOTE: The initial procedures must be ran at least once before any protocol can be applied.

7.1 Ordination procedure

As protocol 3, protocol 4 performs PCoA on a distance matrix calculated with Extended Gower coefficient of dissimilarity, combining Euclidean distances on transformed compositional data (50%) and RRD on ranked petrographic data (50%). In this case, we are not filtering out the shipwreck samples, but we do exclude the true outliers (IND, observations with no group assigned) so they don't pollute visualization.

```
prot4_Shipwreck_2d <- apply_ordination(# no true outliers
                                       cleanAmphorae[!isTrueIND,],
                                       "4", # select protocol 4
                                       exception_columns = excep_cols,
                                       variable_tags = varCode,
                                       coda_override = chemVars16,
                                       coda_transformation = "CLR")

prot4_Shipwreck_3d <- apply_ordination(# no true outliers
                                       cleanAmphorae[!isTrueIND,],
                                       "4", # select protocol 4
                                       exception_columns = excep_cols,
                                       variable_tags = varCode,
                                       coda_override = chemVars16,
                                       coda_transformation = "CLR",
                                       dimensions = 3)
```

7.2 Simplify CoDa names

We may want to simplify the names of the transformed variables before plotting them in a biplot.

```
prot4_Shipwreck_2d <- simplify_coda_names(prot4_Shipwreck_2d)
prot4_Shipwreck_3d <- simplify_coda_names(prot4_Shipwreck_3d)
```

7.3 Test the given provenance groups

We can test the provenance assigned to shipwrecks' amphorae samples together with those found and assigned to the workshops.

```
prot4_Shipwreck_tests <- test_groups(prot4_Shipwreck_2d$dist_matrix,
                                     factor_list_Shipwreck$ProvGroup)
```

These tests were explained in protocol 1.

7.4 Biplots

The details on how to create biplots is already explained in protocol 1. Protocol 4 generates PCoA projections.

7.4.1 Biplot 2D

```
arrows_label_adj <- rbind(c(.5,0),c(.5,1),c(.5,0),c(.5,1),c(.5,0),
                          c(.5,1),c(.8,0),c(1,.5),c(.5,0),c(1,.2),
                          c(.5,1),c(.2,.7))
row.names(arrows_label_adj) <- c("S7","S8","S4","CaO","MgO",
                                 "S11","L48","SiO2","Ce","Nb",
                                 "Th","TiO2")

biplot2d3d::biplot_2d(prot4_Shipwreck_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      ylim = c(-.3,.25),
                      point_type = "point",
                      groups = factor_list_Shipwreck$ProvGroup,
                      group_color = color_list_Shipwreck$ProvGroup,
                      group_label_cex = 0.6,
```

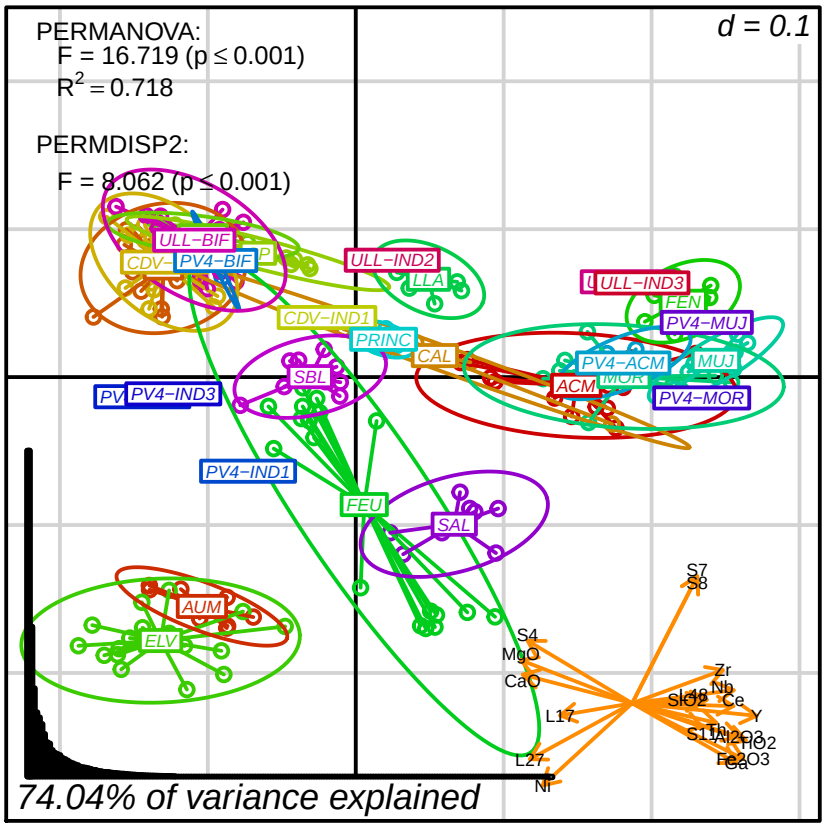


Figure 16: protocol 4 with shipwrecks

```
                    arrow_mim_dist = .5,
                    arrow_label_cex = 0.6,
                    arrow_fig = c(.6,.95,0,.35),
                    arrow_label_adj_override = arrows_label_adj,
                    subtitle = prot4_Shipwreck_2d$sub2D,
                    test_text =
                      prot4_Shipwreck_tests$text(prot4_Shipwreck_tests),
                    test_cex = 0.8,
                    test_fig = c(0, 0.5, 0.62, .99),
                    fitAnalysis_fig = c(0,.7,.05,.5),
                    output_type = "preview")
```

7.4.2 Biplot 3D

```
biplot2d3d::biplot_3d(prot4_Shipwreck_3d,
                      ordination_method = "PCoA",
                      point_type = "point",
                      groups = factor_list_Shipwreck$FabricGroup,
                      group_color = color_list_Shipwreck$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot4_Shipwreck_3d$sub3D,
                      test_text =
                        prot4_Shipwreck_tests$text(prot4_Shipwreck_tests),
                      test_cex = 1.25,
                      test_fig = c(0, 0.5, 0.65, .99),
                      view_zoom = 0.9)

biplot2d3d::animation(directory = directories$prot4_Shipwreck,
                      file_name = "Prot4_Shipwreck_Biplot3D")
```

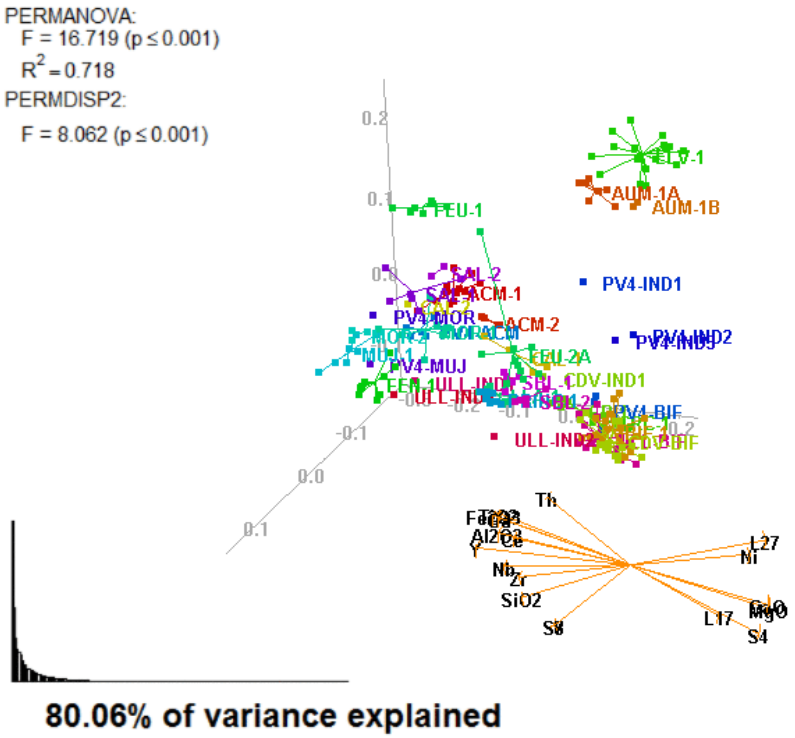


Figure 17: Prot4_Shipwreck_Biplot3D.gif

< back to index

8 Interpreting biplots

This section is a reminder of the possible caveats of interpreting multivariate projections (biplots) as bivariate plots (e.g., scatter plots).

The first big difference between biplots and scatter plots lies in their names. Contrary to common intuition, 'bi' in 'biplot' does not stand for two **axes** or **dimensions** but the two ***plots*** that share the same axes or dimensions. Graphically, those plots consist of points, which is analogous to a scatterplot, and arrows, which represent the covariance between variables and the dimensions of the plot. As these dimensions are given by an ordination method (e.g., PCA), they express the fact that the dataset itself has two dimensions (a matrix with rows and columns). Consequently, three-dimensional biplots are still biplots, not 'triplots'.

There is another, more subtle, difference between biplots and scatter plots. The latter will unequivocally place points according to their values in each of the variables considered. Biplots, in turn, are projections of distributions or 'point clouds' that are multidimensional (i.e., multivariate data). Even in the best scenarios, biplots cannot represent such clouds in their full form. Imagine trying to draw a dice on a sheet of paper.

As an example, consider the outcome of protocol 1. In this case, robust PCA generated a good 2D projection (around 78% of variance) where CaO and MgO are the major contributors.

```
# Recover protocol 1 override for variable label positions
arrows_label_adj <- rbind( c(.5, -.5), c(1, .5), c(1.2, 1.2),
                           c(1.2, .4), c(.8, .5), c(0, 0),
                           c(-.2, 1), c(.5, 1.2), c(-.5, .5),
                           c(-.2, .5), c(0, .5), c(0, 0))
row.names(arrows_label_adj) <- c("Fe2O3", "Al2O3", "SiO2",
                                 "TiO2", "MgO", "Th",
                                 "Nb", "Cr", "Ce",
                                 "Ga", "Zn", "Y")
```

```
biplot_2d(prot1,
          groups = factor_list$ChemGroup,
          group_color = color_list$ChemGroup,
          group_label_cex = 0.6,
          invert_coordinates = c(TRUE, TRUE),
          arrow_label_cex = 0.7,
          test_text = prot1_tests$text(prot1_tests),
          test_cex = 0.8,
          test_fig = c(0, 0.5, 0.65, .99),
          output_type = "preview")
```

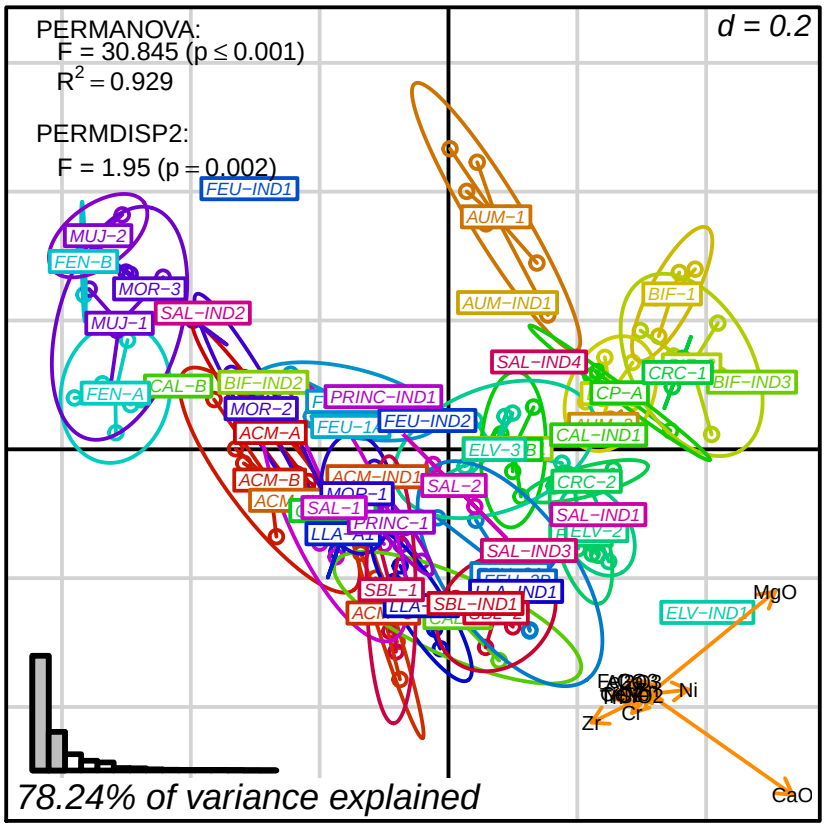


Figure 18: Protocol 1, representing and testing chemical reference groups

Therefore, we can safely interpret positions in terms of having more or less CaO and MgO. For instance, we can classify observations by levels of CaO content and test it against protocol 1 distance matrix and 2D projection:

```
# Create factor variable containing the classification (5 categories)
CaO_level <- cut(cleanAmphorae$CaO[!isShipwreck], 5)

# Select 5 colours from the 'topo.colors' palette
CaO_level_colors <- topo.colors(nlevels(CaO_level))

# Test the classification
prot1_tests_CaO <- test_groups(prot1$dist_matrix, CaO_level)
```

```
# This is for highlighting CaO arrow
arrow_colors <- rep("darkorange", nrow(prot1$loadings))
arrow_colors[row.names(prot1$loadings) == "CaO"] <- "red"

biplot_2d(prot1,
          groups = CaO_level,
          group_color = CaO_level_colors,
          group_star_cex = 0,
          group_label_cex = 0,
          show_group_legend = TRUE,
          group_legend_title = "CaO",
          group_legend_title_pos = c(0.5,0.9),
          group_legend_text_cex = 0.8,
          group_legend_fig = c(0.7,0.99,0.68,0.95),
          invert_coordinates = c(TRUE, TRUE),
          arrow_label_cex = 0.8,
          arrow_fig = c(.6,.95,0,.35),
          arrow_label_adj_override = arrows_label_adj,
          arrow_color = arrow_colors,
          test_text = prot1_tests_CaO$text(prot1_tests_CaO),
          test_cex = 0.8,
          test_fig = c(0, 0.5, 0.65, .99),
          output_type = "preview")
```

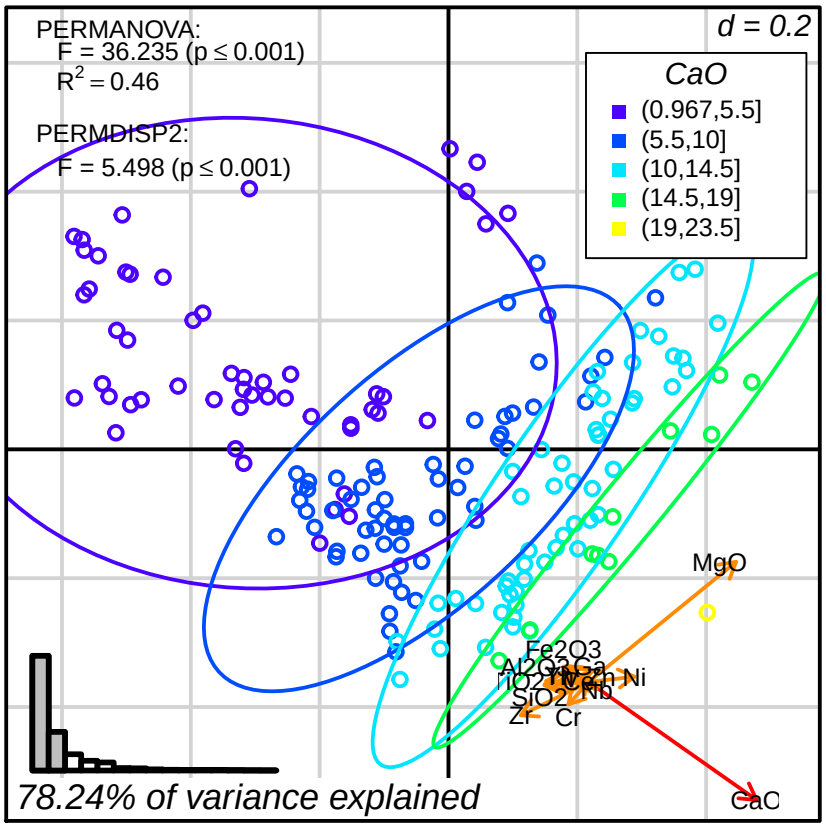


Figure 19: Protocol 1, grouping by level of CaO content

However, interpretation is less straightforward when more than two variables contribute significantly to the total variation. Regarding biplots, such situation implies that a smaller portion of variation is represented, and that several variables are stretch on many directions over the two principal coordinates.

For example, protocol 2 gave us a much worse 2D projection (55.7%) where fifteen variables are well represented.

```
# Recover protocol 2a override for variable label positions
arrows_label_adj <- rbind(c(.5,.8),c(.5,1),c(.5,1),c(.5,0),c(.5,1),
                          c(.5,0),c(0,.5))
row.names(arrows_label_adj) <- c("L48","L24","L5","L36","S7",
                                 "S8","S11")

# This will help us select different arrow colours
isDisplayed <-
  row.names(prot2a_2d$loadings) %in% row.names(
    filter_arrows(prot2a_2d$loadings, min_dist = 0.5))
```

```
biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = 0.5,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot2a_2d$sub2D,
                      test_text = prot2a_tests$text(prot2a_tests),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      output_type = "preview")
```

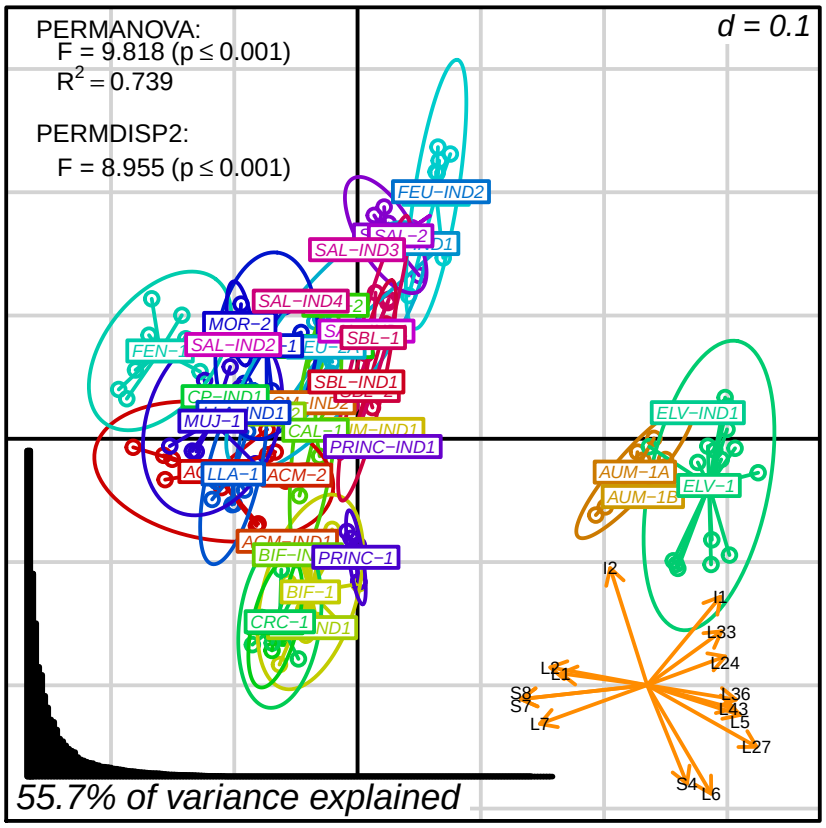


Figure 20: protocol 2a, representing and testing fabric groups

Like in protocol 1, we may want to interpret this projection in terms of a single variable. A obvious candidate is I2 since it is displayed long and quite isolated from other variables.

In this case, I2 (or INCLUS_ORIENT) is already a factor variable (classification) with 3 categories (plus "none" as a missing value). Note that the selected dataset will have cases in only two of those categories.

```
# You may want to assure that the true
# categories are corectly represented:
cleanAmphorae <- order_petro(cleanAmphorae)

levels(cleanAmphorae$INCLUS_ORIENT[!isShipwreck])

# Declare this factor separately as an object for clearness
I2 <- cleanAmphorae$INCLUS_ORIENT[!isShipwreck]

# Select colours from the 'topo.colors' palette
I2_colors <- topo.colors(nlevels(I2))

# Test the classification
prot1_tests_I2 <- test_groups(prot2a_2d$dist_matrix, I2)
```

```r
# This is for highlighting I2 arrow
arrow_colors <- rep("darkorange", nrow(prot2a_2d$loadings))
arrow_colors[row.names(prot2a_2d$loadings) == "I2"] <- "red"
# filter arrows colours, since not all variables are displayed
arrow_colors <- arrow_colors[isDisplayed]

biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
                      groups = I2,
                      group_color = I2_colors,
                      group_star_cex = 0,
                      group_label_cex = 0,
                      show_group_legend = TRUE,
                      group_legend_title = "INCLUS_ORIENT",
                      group_legend_title_pos = c(0.5,0.9),
                      group_legend_text_cex = 0.8,
                      group_legend_fig = c(0.6,0.99,0.68,0.95),
                      arrow_mim_dist = .5,
                      arrow_label_cex = 0.8,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      arrow_color = arrow_colors,
                      subtitle = prot2a_2d$sub2D,
                      test_text = prot1_tests_I2$text(prot1_tests_I2),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      output_type = "preview")
```

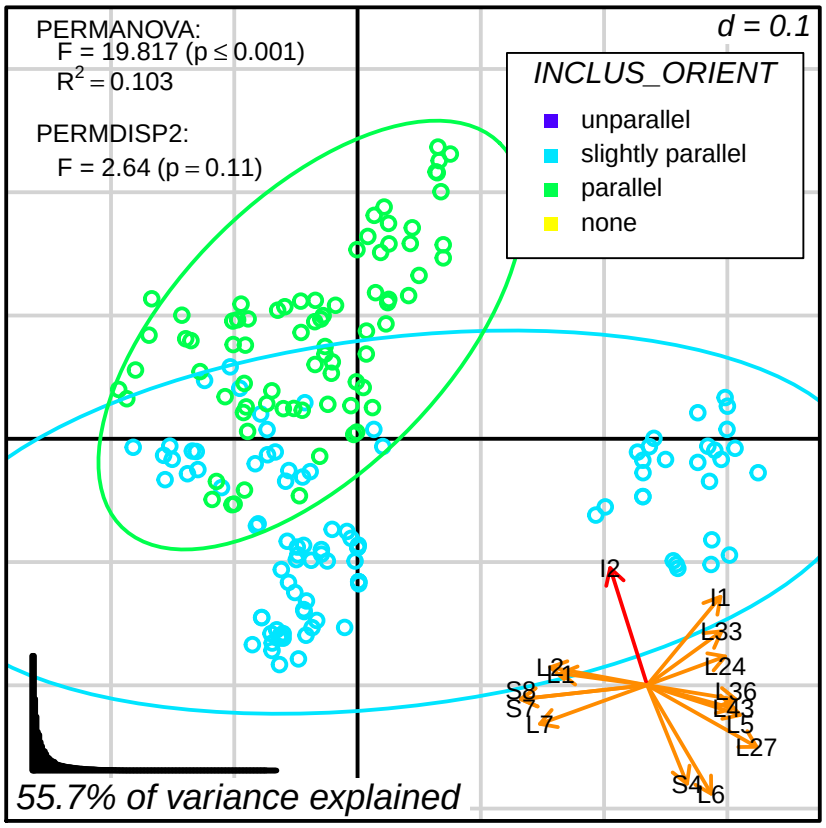


Figure 21: Protocol 2a, grouping by INCLUS_ORIENT

This kind of reading becomes increasingly difficult when focusing in variables that are not so well aligned, such as L33 (or COAR_R_CHERT)

```
# Declare this factor separately as an object for clearness
L33 <- cleanAmphorae$COAR_R_CHERT[!isShipwreck]

# Select colours from the 'topo.colors' palette
L33_colors <- topo.colors(nlevels(L33))

# Test the classification
prot1_tests_L33 <- test_groups(prot2a_2d$dist_matrix, L33)
```

```r
# This is for highlighting L33 arrow
arrow_colors <- rep("darkorange", nrow(prot2a_2d$loadings))
arrow_colors[row.names(prot2a_2d$loadings) == "L33"] <- "red"
# filter arrows colours, since not all variables are displayed
arrow_colors <- arrow_colors[isDisplayed]

biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
                      groups = L33,
                      group_color = L33_colors,
                      group_star_cex = 0,
                      group_label_cex = 0,
                      show_group_legend = TRUE,
                      group_legend_title = "COAR_R_CHERT",
                      group_legend_title_pos = c(0.5,0.9),
                      group_legend_text_cex = 0.8,
                      group_legend_fig = c(0.6,0.99,0.68,0.95),
                      arrow_mim_dist = .5,
                      arrow_label_cex = 0.8,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      arrow_color = arrow_colors,
                      subtitle = prot2a_2d$sub2D,
                      test_text = prot1_tests_L33$text(prot1_tests_L33),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      output_type = "preview")
```

Biplots and ordinal methods (PCA, PCoA, CA, etc.) are exploratory tools that play a game of compromise in order to define the best projections given the whole variance in a dataset. Do not expect them to display patterns that are clear when looking into specific variables. For that kind of analysis, you should use bivariate statistics and graphical displays, such as scatter plots or box plots.

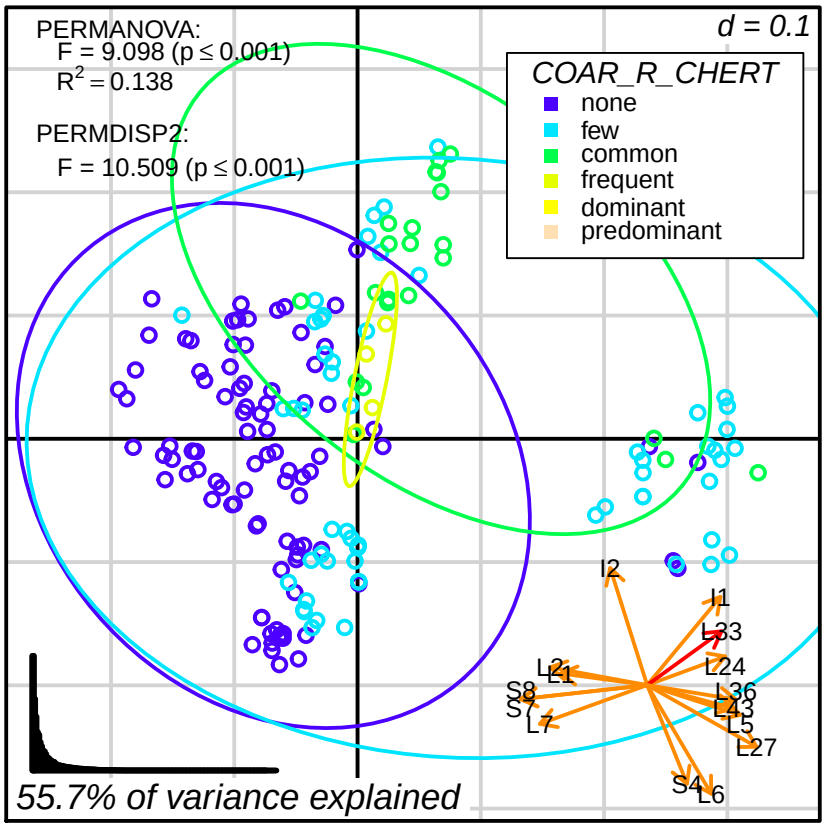


Figure 22: Protocol 2a, grouping by COAR_R_CHERT

< back to index

5.3.4.1. Appendix D: Scripts

```
[1] "Esto es la impresión integral del archivo 'Initial_procedures.R'"
```

```
# License notice:
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#
############################################################
# cerUB - Initial procedures (preparing for protocols)
# This script should be executed before applying a cerUB protocol
############################################################

library(cerUB)

# Directories ------------------------------------------------------------

directories <- list(
  data = "data",
  transCoDa = "transformed CoDa",

  prot1 = "Protocol_1_geochemical_data",
  prot2 = "Protocol_2_petrographic_data",
  prot3 = "Protocol_3_geochemical_and_petrographic_data",
  prot4 = "Protocol_4_provenance_data",
  prot4_Shipwreck = "Protocols_4_provenance_data_with_shipwrecks"
```

```
)

lapply(directories, dir.create, showWarnings = FALSE)


# Read data ------------------------------------------------------------

data(amphorae)

# alternativelly, using your own data (e.g., CSV)
# dt <- cbind(read.csv(paste(directories$data,
#                            "petrographic_data.csv",sep="/"),
#                      row.names=1), # assuming the first column contains row names
#             read.csv(paste(directories$data,
#                            "geochemical_data.csv",sep="/"),
#                      row.names=1)) # assuming the first column contains row names


# Codify petrographical variables --------------------------------------

varCode <- code_variables(amphorae)


# Clean and format data ------------------------------------------------

cleanAmphorae <- clean_and_format(amphorae,
                                  completion_variable = c("CHARAC", "complete"),
                                  categorical_columns = 1:112,
                                  numerical_columns = 113:ncol(amphorae),
                                  as_na = c("NULL", "indeterminate", "unfired"),
                                  method = NULL,
                                  # don't use the following variables
                                  columns_to_exclude = c("VOID_VESIC_MEGA",
                                                         "VOID_VUGH_MEGA",
```

```
                                                    "VOID_CHAN_MEGA",
                                                    "VOID_PLAN_MEGA",
                                                    "COAR_R_DAC_AND",
                                                    "COAR_R_EVAP",
                                                    "COAR_R_CONGBREC",
                                                    "COAR_R_SERP",
                                                    "COAR_C_SPL",
                                                    "COAR_C_OPX",
                                                    "COAR_C_OL",
                                                    "COAR_C_SIL",
                                                    "COAR_C_ST",
                                                    "COAR_C_ZRN",
                                                    "COAR_C_PY",
                                                    "FINE_C_OPX",
                                                    "FINE_C_ZRN"),
                                    # don't use the following observations
                                    rows_to_exclude = c("PV4033", # PV4-IND4
                                                        "PV4017", # PV4-CAMP
                                                        # PV4-ITT
                                                        "PV4021", "PV4023",
                                                        "PV4024", "PV4025",
                                                        "PV4035", "PV4037",
                                                        # PV4-NAP
                                                        "PV4022", "PV4026",
                                                        "PV4027", "PV4028",
                                                        "PV4029", "PV4030",
                                                        "PV4036")
)


# Subsetting criteria ------------------------------------------------

# Build vector indicating wheter each observation is from a shipwreck
isShipwreck <-
```

```
  cleanAmphorae$Site_Name=="Cap del Vol" |
  cleanAmphorae$Site_Name=="Ullastres I" |
  cleanAmphorae$Site_Name=="Port-Vendres 4"


# Build vectors indicating provenance group and whether observations are
# true outliers (IND). Also, reformat FabricGroup and ChemReferenceGroup,
# so true outliers are single out separatelly and not as a extra group.
ProvenanceGroup <- c()
isTrueIND <- c()

# coerce the original group variables (factors) into character vectors
# so we can use stringr package to operate on them.
cleanAmphorae$FabricGroup <- as.character(cleanAmphorae$FabricGroup)
cleanAmphorae$ChemReferenceGroup <- as.character(cleanAmphorae$ChemReferenceGroup)

for (i in 1:nrow(cleanAmphorae)){

  groupChem <-
    stringr::str_split(cleanAmphorae$ChemReferenceGroup[i], "-")[[1]]
  groupFabric <-
    stringr::str_split(cleanAmphorae$FabricGroup[i], "-")[[1]]
  group <- ""
  isATrueInd <- FALSE

  if (groupChem[2] == "IND" || groupFabric[2] == "IND") {

    group <- cleanAmphorae$ChemReferenceGroup[i]

    if (!isShipwreck[i]) isATrueInd <- TRUE

    index <- 1
    for (j in 1:length(ProvenanceGroup)){
```

```
      if (ProvenanceGroup[j] == paste(group, index, sep = ""))
        index <- index + 1

    }

    group <- paste(group, index, sep = "")
    cleanAmphorae$ChemReferenceGroup[i] <- group
    cleanAmphorae$FabricGroup[i] <- group
  }
  else {
    if (groupChem[1] == "ULL" || groupChem[1] == "PV4" || groupChem[1] == "CDV"){
      group <- cleanAmphorae$ChemReferenceGroup[i]
    }
    else if (groupChem[1] == groupFabric[1]){
      group <- groupChem[1]
    }
  }
  ProvenanceGroup <- c(ProvenanceGroup, group[1])
  isTrueIND <- c(isTrueIND, isATrueInd)
}


# Build lists of named group factors for easiness of reference ------------

# list aiming to define workshops productions, so no shipwrecks
factor_list <-
  list(
    Site = factor(cleanAmphorae$Site_Name[!isShipwreck]),
    FabricGroup = factor(cleanAmphorae$FabricGroup[!isShipwreck]),
    ChemGroup = factor(cleanAmphorae$ChemReferenceGroup[!isShipwreck]),
    ProvGroup = factor(ProvenanceGroup[!isShipwreck])
  )

# list aiming to assign shipwreck observations to workshop productions,
```

```
# so with shipwreck samples but no true outliers
factor_list_Shipwreck <-
  list(
    Site = factor(cleanAmphorae$Site_Name[!isTrueIND]),
    FabricGroup = factor(cleanAmphorae$FabricGroup[!isTrueIND]),
    ChemGroup = factor(cleanAmphorae$ChemReferenceGroup[!isTrueIND]),
    ProvGroup = factor(ProvenanceGroup[!isTrueIND])
  )



# Build lists of named point types vectors for easiness of reference -------------

# point type full vectors
labels_code <- as.character(row.names(cleanAmphorae))
labels_cross <- rep("+", nrow(cleanAmphorae))
labels_x <- rep(4, nrow(cleanAmphorae)) # using pch code
labels_point <- rep(20, nrow(cleanAmphorae)) # using pch code

# list aiming to define workshops productions, so no shipwrecks
labels_list <- list(
  Code = labels_code[!isShipwreck],
  Cross = labels_cross[!isShipwreck],
  X = labels_x[!isShipwreck],
  Point = labels_point[!isShipwreck]
)

# list aiming to assign shipwreck observations to workshop productions,
# so with shipwreck samples but no true outliers
labels_list_Shipwreck <- list(
  Code = labels_code[!isTrueIND],
  Cross = labels_cross[!isTrueIND],
  X = labels_x[!isTrueIND],
  Point = labels_point[!isTrueIND]
)
```

```
# Build lists of named group color vectors for easiness of reference ------------

color_list <- list()
for (i in 1:length(factor_list)){
  cv <- rainbow(nlevels(factor_list[[i]]), v=.8)
  color_list[[i]] <- cv
  names(color_list)[i] = names(factor_list)[i]
}

color_list_Shipwreck <- list()
for (i in 1:length(factor_list_Shipwreck)){
  cv <- rainbow(nlevels(factor_list_Shipwreck[[i]]), v=.8)
  color_list_Shipwreck[[i]] <- cv
  names(color_list_Shipwreck)[i] = names(factor_list_Shipwreck)[i]
}


# Enunciate exception columns ----------------------------------------------

# enunciates which ordinal variables have "none" as a exceptional value
# when calculating the distance between values.

excep_cols <- c("INCLUS_DISTRIB","INCLUS_ORIENT","COAR_ROUNDNESS",
                "COAR_FORM","COAR_SPACING","COAR_SORTING","FINE_FORM")


# Check order of petrographic variables -----------------------------------

# no need to save it, because apply_protocol will do it internally
str(order_petro(cleanAmphorae))
```

```
# Select (and save transformed geochemical data) ---------------------------

chemVars16 <- c("Fe2O3","Al2O3","TiO2","MgO","CaO","SiO2",
                "Th","Nb","Zr","Y","Ce","Ga","V","Zn","Ni","Cr")

# Save transformed CoDa to file (optional)
# no need to save it in the environment, because apply_protocol will transform
# the data internally and save the results in ordination_object$logratio_data,
# when applicable
# NOTE: In the output table, CoDa columns will be ordered as:
# (1) CoDa variables not transformed,
# (2) Raw version of the selected CoDa variables,
# (3) Transformed version of the selected CoDa variables.

write(transform_coda(cleanAmphorae,
                     coda_variables = chemVars16,
                     method = c("CLR")),
      file = paste(directories$transCoDa, "transAmphorae_clr.csv", sep = "/"))
```

```
[1] "Esto es la impresión integral del archivo 'Protocol_1.R'"

# License notice:
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

```
#
############################################################
# cerUB - Protocol 1
#
# 1. geochemical compositional data (CoDa)
# 2. isometric log-ratio transformation (ilr)
# 3. robust Principal Components Analysis (robPCA), implicitly using Euclidean distanc
# 4. PERMANOVA & PERMDISP tests
# 5. (outlier detection)
#
############################################################


# NOTE: "Initial procedures.R" must be ran once first.



# Apply protocol 1 to confirm workshops' chemical reference groups --------

prot1 <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                          "1", # select protocol 1
                          coda_override = chemVars16,
                          coda_transformation = "ILR")

# Simplify CoDa names for plot clearness
prot1 <- simplify_coda_names(prot1)


# Test the given chemical reference groups --------------------------------

# Note: This test batch may take several minutes depending on the size of the
# data matrix and the number of groups.
prot1_tests <- test_groups(prot1$dist_matrix, factor_list$ChemGroup)
```

```
# Biplot 2D ----------------------------------------------------------------

arrows_label_adj <- rbind( c(.5, -.5), c(1, .5), c(1.2, 1.2), c(1.2, .4),
                           c(.8, .5),
                           c(0, 0), c(-.2, 1), c(.5, 1.2), c(-.5, .5),
                           c(-.2, .5), c(0, .5), c(0, 0))
row.names(arrows_label_adj) <- c("Fe2O3", "Al2O3", "SiO2", "TiO2",
                                 "MgO",
                                 "Th", "Nb", "Cr", "Ce",
                                 "Ga", "Zn", "Y")

# better "preview" (R UI device) version
biplot2d3d::biplot_2d(prot1,
                      groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      group_ellipse_cex = 2.5,
                      group_label_cex = 0.6,
                      invert_coordinates = c(FALSE, TRUE),
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      test_text = prot1_tests$text(prot1_tests),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      output_type = "preview")

# better PNG version
biplot2d3d::biplot_2d(prot1,
                      ordination_method = "PCA",
                      invert_coordinates = c (TRUE,TRUE),
                      grid_cex = 2.5,
                      ylim = c(-.1,.1),
                      point_type = "point",
```

```
                        groups = factor_list$ChemGroup,
                        group_color = color_list$ChemGroup,
                        group_label_cex = 1.5,
                        arrow_label_cex = 2,
                        arrow_cex = 0.2,
                        arrow_lwd = 2.5,
                        arrow_fig = c(.6,.95,0,.35),
                        arrow_label_adj_override = arrows_label_adj,
                        subtitle = prot1$sub2D,
                        subtitle_cex = 2.5,
                        test_text = prot1_tests$text(prot1_tests),
                        test_spacing_line = 0.9,
                        test_fig =c(0, 0.5, 0.72, .99),
                        test_cex = 2,
                        fitAnalysis_fig = c(0,.7,.05,.5),
                        file_name = "Prot1_Biplot2D",
                        directory = directories$prot1,
                        width = 1000, height = 1000,
                        output_type = "png")

# better EPS version
biplot2d3d::biplot_2d(prot1,
                        ordination_method = "PCA",
                        invert_coordinates = c (TRUE,TRUE),
                        grid_cex = 2.5,
                        ylim = c(-.1,.1),
                        point_type = "point",
                        groups = factor_list$ChemGroup,
                        group_color = color_list$ChemGroup,
                        group_label_cex = 1.5,
                        arrow_label_cex = 1.5,
                        arrow_cex = 0.2,
                        arrow_lwd = 2.5,
                        arrow_fig = c(.6,.95,0,.35),
```

```
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot1$sub2D,
                      subtitle_cex = 2.5,
                      test_text = prot1_tests$text(prot1_tests),
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      test_cex = 1.5,
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      file_name = "Prot1_Biplot2D",
                      directory = directories$prot1,
                      width = 1000, height = 1000,
                      output_type = c("eps"))

# Biplot 3D ------------------------------------------------------------------

biplot2d3d::biplot_3d(prot1,
                      ordination_method = "PCA",
                      groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      point_type = "point",
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      test_text = prot1_tests$text(prot1_tests),
                      test_cex = 1.25,
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99))

# save animated GIF or PNG snapshot
biplot2d3d::animation(file_name = "Prot1_Biplot3D")
```

```
[1] "Esto es la impresión integral del archivo 'Protocol_2.R'"

# License notice:
```

```
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#
############################################################
# cerUB - Protocol 2
#
# 1. Ordinal petrographic data
# 2. transformed to ranks
# 3. (a) Relative ranking
#    (b) Neighbor interchange
# 4. (a) Principal Coordinates Analysis (PCoA)
#    (b) Non-metric Dimensional Scaling (NMDS)
# 4. PERMANOVA & PERMDISP tests
#
############################################################


# NOTE: "Initial procedures.R" must be ran once first.


# Apply protocol 2 to identify workshops ----------------------------------

# project in 2d (default)
```

```
prot2a_2d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                              "2a", # select protocol 2a (RRD & PCoA)
                              exception_columns = excep_cols,
                              variable_tags = varCode)

prot2b_2d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                              "2b", # select protocol 2a (NI & NMDS)
                              exception_columns = excep_cols,
                              variable_tags = varCode)

# project in 3d (dimensions = 3)

prot2a_3d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                              "2a", # select protocol 2a (RRD & PCoA)
                              exception_columns = excep_cols,
                              variable_tags = varCode,
                              dimensions = 3)

prot2b_3d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                              "2b", # select protocol 2a (NI & NMDS)
                              exception_columns = excep_cols,
                              variable_tags = varCode,
                              dimensions = 3)

# Test the given fabric groups ------------------------------

# Note1: Tests depend only on the distance matrix and the groups, not on the projectio
# Note2: This test batch may take several minutes depending on the size of the
# data matrix and the number of groups.
prot2a_tests <- test_groups(prot2a_2d$dist_matrix,
                            factor_list$FabricGroup)
prot2b_tests <- test_groups(prot2b_2d$dist_matrix,
                            factor_list$FabricGroup)
```

```
# Biplot 2D ----------------------------------------------------------

## prot2a
arrows_label_adj <- rbind(c(.5,.8),c(.5,1),c(.5,1),c(.5,0),c(.5,1),c(.5,0),c(0,.5))
row.names(arrows_label_adj) <- c("L48","L24","L5","L36","S7","S8","S11")

# better "preview" (R UI device) version
biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = 0,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot2a_2d$sub2D,
                      test_text = prot2a_tests$text(prot2a_tests),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      output_type = "preview")

# better PNG version
biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      grid_cex = 2.5,
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
```

```
                        point_type = "point",
                        groups = factor_list$FabricGroup,
                        group_color = color_list$FabricGroup,
                        group_label_cex = 1.5,
                        arrow_mim_dist = .5,
                        arrow_label_cex = 2,
                        arrow_cex = 0.2,
                        arrow_lwd = 2.5,
                        arrow_fig = c(.6,.95,0,.35),
                        arrow_label_adj_override = arrows_label_adj,
                        subtitle = prot2a_2d$sub2D,
                        subtitle_cex = 2.5,
                        test_text = prot2a_tests$text(prot2a_tests),
                        test_spacing_line = 0.9,
                        test_fig =c(0, 0.5, 0.72, .99),
                        test_cex = 2,
                        fitAnalysis_fig = c(0,.7,.05,.5),
                        file_name = "Prot2a_Biplot2D",
                        directory = directories$prot2,
                        width = 1000, height = 1000,
                        output_type = "png")

# better EPS version
biplot2d3d::biplot_2d(prot2a_2d,
                        ordination_method = "PCoA",
                        invert_coordinates = c (TRUE,TRUE),
                        grid_cex = 2.5,
                        xlim = c(-.26,.35),
                        ylim = c(-.31,.35),
                        point_type = "point",
                        groups = factor_list$FabricGroup,
                        group_color = color_list$FabricGroup,
                        group_label_cex = 1.5,
                        arrow_mim_dist = .5,
```

```
                        arrow_label_cex = 1.5,
                        arrow_cex = 0.2,
                        arrow_lwd = 2.5,
                        arrow_fig = c(.6,.95,0,.35),
                        arrow_label_adj_override = arrows_label_adj,
                        subtitle = prot2a_2d$sub2D,
                        subtitle_cex = 2.5,
                        test_text = prot2a_tests$text(prot2a_tests),
                        test_spacing_line = 0.9,
                        test_fig =c(0, 0.5, 0.72, .99),
                        test_cex = 1.5,
                        fitAnalysis_fig = c(0,.7,.05,.5),
                        file_name = "Prot2a_Biplot2D",
                        directory = directories$prot2,
                        width = 1000, height = 1000,
                        output_type = "eps")

## prot2b

arrows_label_adj <- rbind(c(.5,1),c(.5,0),c(.5,1),c(.5,1),c(.5,0),c(0,.5),c(1,.5))
row.names(arrows_label_adj) <- c("S7","S8","CLAY","L24","L43","L5","L36")

# better "preview" (R UI device) version
biplot2d3d::biplot_2d(prot2b_2d,
                        ordination_method = "NMDS",
                        xlim = c(-.42,.38),
                        ylim = c(-.45,.25),
                        point_type = "point",
                        groups = factor_list$FabricGroup,
                        group_color = color_list$FabricGroup,
                        group_label_cex = 0.6,
                        arrow_mim_dist = .5,
                        arrow_label_cex = 0.6,
                        arrow_fig = c(.6,.95,0,.35),
```

```
                        arrow_label_adj_override = arrows_label_adj,
                        subtitle = prot2b_2d$sub2D,
                        test_text = prot2b_tests$text(prot2b_tests),
                        test_cex = 0.8,
                        test_fig = c(0, 0.5, 0.65, .99),
                        fitAnalysis_fig = c(.1,.6,.1,.4),
                        output_type = "preview")

# better PNG version
biplot2d3d::biplot_2d(prot2b_2d,
                        ordination_method = "NMDS",
                        grid_cex = 2.5,
                        xlim = c(-.42,.38),
                        ylim = c(-.45,.25),
                        point_type = "point",
                        groups = factor_list$FabricGroup,
                        group_color = color_list$FabricGroup,
                        group_label_cex = 1.5,
                        arrow_mim_dist = .5,
                        arrow_label_cex = 2,
                        arrow_cex = 0.2,
                        arrow_lwd = 2.5,
                        arrow_fig = c(.6,.95,0,.35),
                        arrow_label_adj_override = arrows_label_adj,
                        subtitle = prot2b_2d$sub2D,
                        subtitle_cex = 2.5,
                        test_text = prot2b_tests$text(prot2b_tests),
                        test_spacing_line = 0.9,
                        test_fig =c(0, 0.5, 0.72, .99),
                        test_cex = 2,
                        fitAnalysis_fig = c(.1,.6,.1,.4),
                        file_name = "Prot2b_Biplot2D",
                        directory = directories$prot2,
                        width = 1000, height = 1000,
```

```
                      output_type = "png")

# better EPS version
biplot2d3d::biplot_2d(prot2b_2d,
                      ordination_method = "NMDS",
                      grid_cex = 2.5,
                      xlim = c(-.42,.38),
                      ylim = c(-.45,.25),
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_label_cex = 1.5,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 1.5,
                      arrow_cex = 0.2,
                      arrow_lwd = 2.5,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot2b_2d$sub2D,
                      subtitle_cex = 2.5,
                      test_text = prot2b_tests$text(prot2b_tests),
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      test_cex = 1.5,
                      fitAnalysis_fig = c(.1,.6,.1,.4),
                      file_name = "Prot2b_Biplot2D",
                      directory = directories$prot2,
                      width = 1000, height = 1000,
                      output_type = "eps")

# Biplot 3D ------------------------------------------------------------------

## prot2a
```

```
biplot2d3d::biplot_3d(prot2a_3d,
                      ordination_method = "PCoA",
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot2a_3d$sub3D,
                      test_text = prot2a_tests$text(prot2a_tests),
                      test_cex = 1.25,
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      view_zoom = 0.9)

# save animated GIF or PNG snapshot
biplot2d3d::animation(directory = directories$prot2,
                       file_name = "Prot2a_Biplot3D")


biplot2d3d::biplot_3d(prot2b_3d,
                      ordination_method = "NMDS",
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot2b_3d$sub3D,
                      test_text = prot2b_tests$text(prot2b_tests),
```

```
                      test_cex = 1.25,
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      view_zoom = 0.9)

# save animated GIF or PNG snapshot
biplot2d3d::animation(directory = directories$prot2,
                       file_name = "Prot2b_Biplot3D")
```

```
[1] "Esto es la impresión integral del archivo 'Protocol_3.R'"

# License notice:
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#
############################################################
# cerUB - Protocol 3
#
# 1. geochemical compositional data (CoDa) & Ordinal petrographic data
# 2. centred log-ratio transformation (clr) & transformed to ranks
# 3. Extended Gower Coefficient: Euclidean distance & Relative ranking
# 4. Principal Coordinates Analysis (PCoA)
# 4. PERMANOVA & PERMDISP tests
#
```

```
############################################################


# NOTE: "Initial procedures.R" must be ran once first.


# Apply protocol 3 to identify workshops ---------------------------------

# project in 2d (default)

prot3_2d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                             "3", # select protocol 3
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR")

# Simplify CoDa names for plot clearness
prot3_2d <- simplify_coda_names(prot3_2d)


# project in 3d (dimensions = 3)

prot3_3d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                             "3", # select protocol 3
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR",
                             dimensions = 3)

# Simplify CoDa names for plot clearness
prot3_3d <- simplify_coda_names(prot3_3d)
```

```
# Test the given provenance groups -------------------------------

# Note1: Tests depend only on the distance matrix and the groups, not on the projectio
# Note2: This test batch may take several minutes depending on the size of the
# data matrix and the number of groups.
prot3_tests <- test_groups(prot3_2d$dist_matrix,
                           factor_list$ChemGroup)

# Biplot 2D ------------------------------------------------------------------

arrows_label_adj <- rbind(c(0,0),c(.5,0),c(.8,.2),c(.5,0),c(.5,1),c(.5,1.1),c(1,.5),
                          c(0,.5))
row.names(arrows_label_adj) <- c("L6","Ga","S11","S7","S8","Nb","SiO2",
                                 "Y")

# better "preview" (R UI device) version
biplot2d3d::biplot_2d(prot3_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (FALSE,FALSE),
                      ylim = c(-.3,.29),
                      point_type = "point",
                      groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot3_2d$sub2D,
                      test_text = prot3_tests$text(prot3_tests),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      output_type = "preview")
```

```
# better PNG version
biplot2d3d::biplot_2d(prot3_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      grid_cex = 2.5,
                      ylim = c(-.1,.1),
                      point_type = "point",
                      groups = factor_list$ChemGroup,
                      group_color = color_list$ChemGroup,
                      group_label_cex = 1.5,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 2,
                      arrow_cex = 0.2,
                      arrow_lwd = 2.5,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot3_2d$sub2D,
                      subtitle_cex = 2.5,
                      test_text = prot3_tests$text(prot3_tests),
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      test_cex = 2,
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      file_name = "Prot3_Biplot2D",
                      directory = directories$prot3,
                      width = 1000, height = 1000,
                      output_type = "png")

# better EPS version
biplot2d3d::biplot_2d(prot3_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      grid_cex = 2.5,
```

```
                        ylim = c(-.1,.1),
                        point_type = "point",
                        groups = factor_list$ChemGroup,
                        group_color = color_list$ChemGroup,
                        group_label_cex = 1.5,
                        arrow_mim_dist = .5,
                        arrow_label_cex = 1.5,
                        arrow_cex = 0.2,
                        arrow_lwd = 2.5,
                        arrow_fig = c(.6,.95,0,.35),
                        arrow_label_adj_override = arrows_label_adj,
                        subtitle = prot3_2d$sub2D,
                        subtitle_cex = 2.5,
                        test_text = prot3_tests$text(prot3_tests),
                        test_spacing_line = 0.9,
                        test_fig =c(0, 0.5, 0.72, .99),
                        test_cex = 1.5,
                        fitAnalysis_fig = c(0,.7,.05,.5),
                        file_name = "Prot3_Biplot2D",
                        directory = directories$prot3,
                        width = 1000, height = 1000,
                        output_type = "eps")

# Biplot 3D ------------------------------------------------------------------

biplot2d3d::biplot_3d(prot3_3d,
                        ordination_method = "PCoA",
                        point_type = "point",
                        groups = factor_list$ChemGroup,
                        group_color = color_list$ChemGroup,
                        group_representation = "stars",
                        star_centroid_radius = 0,
                        star_label_cex = .8,
                        arrow_min_dist = .5,
```

```
                    arrow_body_length = .025,
                    subtitle = prot3_3d$sub3D,
                    test_text = prot3_tests$text(prot3_tests),
                    test_cex = 1.25,
                    test_spacing_line = 0.9,
                    test_fig =c(0, 0.5, 0.72, .99),
                    view_zoom = 0.9)

# save animated GIF or PNG snapshot
biplot2d3d::animation(directory = directories$prot3,
                      file_name = "Prot3_Biplot3D")
```

```
[1] "Esto es la impresión integral del archivo 'Protocol_4.R'"

# License notice:
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#
############################################################
# cerUB - Protocol 4
#
# 1. geochemical compositional data (CoDa) & Ordinal petrographic data (Provenance sel
# 2. centred log-ratio transformation (clr) & transformed to ranks
# 3. Extended Gower Coefficient: Euclidean distance & Relative ranking
```

```
# 4. Principal Coordinates Analysis (PCoA)
# 4. PERMANOVA & PERMDISP tests
#
############################################################


# NOTE: "Initial procedures.R" must be ran once first.


# Apply protocol 4 to identify workshops ---------------------------------

# project in 2d (default)

prot4_2d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                             "4", # select protocol 4
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR")

# Simplify CoDa names for plot clearness
prot4_2d <- simplify_coda_names(prot4_2d)


# project in 3d (dimensions = 3)

prot4_3d <- apply_ordination(cleanAmphorae[!isShipwreck,], # no shipwrecks
                             "4", # select protocol 4
                             exception_columns = excep_cols,
                             variable_tags = varCode,
                             coda_override = chemVars16,
                             coda_transformation = "CLR",
                             dimensions = 3)
```

```
# Simplify CoDa names for plot clearness
prot4_3d <- simplify_coda_names(prot4_3d)

# Test the given provenance groups --------------------------------

# Note1: Tests depend only on the distance matrix and the groups, not on the projectio
# Note2: This test batch may take several minutes depending on the size of the
# data matrix and the number of groups.
prot4_tests <- test_groups(prot4_2d$dist_matrix,
                           factor_list$ProvGroup)

# Biplot 2D -------------------------------------------------------------

arrows_label_adj <- rbind(c(.5,1),c(0,0),c(1,.5),c(0,1),c(1,0),c(0,.5),c(.5,1),
                          c(1,.5),c(.5,1))
row.names(arrows_label_adj) <- c("CaO","S4","S7","S8","Ce","Nb","Al2O3",
                                 "S11","Fe2O3")

# better "preview" (R UI device) version
biplot2d3d::biplot_2d(prot4_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,FALSE),
                      ylim = c(-.35,.32),
                      point_type = "point",
                      groups = factor_list$ProvGroup,
                      group_color = color_list$ProvGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot4_2d$sub2D,
                      test_text = prot4_tests$text(prot4_tests),
                      test_cex = 0.8,
```

```
                      test_fig = c(0, 0.5, 0.62, .99),
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      output_type = "preview")

# better PNG version
biplot2d3d::biplot_2d(prot4_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,FALSE),
                      grid_cex = 2.5,
                      ylim = c(-.3,.25),
                      point_type = "point",
                      groups = factor_list$ProvGroup,
                      group_color = color_list$ProvGroup,
                      group_label_cex = 1.5,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 2,
                      arrow_cex = 0.2,
                      arrow_lwd = 2.5,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot4_2d$sub2D,
                      subtitle_cex = 2.5,
                      test_text = prot4_tests$text(prot4_tests),
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      test_cex = 2,
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      file_name = "Prot4_Biplot2D",
                      directory = directories$prot4,
                      width = 1000, height = 1000,
                      output_type = "png")

# better EPS version
biplot2d3d::biplot_2d(prot4_2d,
```

```
                        ordination_method = "PCoA",
                        invert_coordinates = c (TRUE,FALSE),
                        grid_cex = 2.5,
                        ylim = c(-.3,.25),
                        point_type = "point",
                        groups = factor_list$ProvGroup,
                        group_color = color_list$ProvGroup,
                        group_label_cex = 1.5,
                        arrow_mim_dist = .5,
                        arrow_label_cex = 1.5,
                        arrow_cex = 0.2,
                        arrow_lwd = 2.5,
                        arrow_fig = c(.6,.95,0,.35),
                        arrow_label_adj_override = arrows_label_adj,
                        subtitle = prot4_2d$sub2D,
                        subtitle_cex = 2.5,
                        test_text = prot4_tests$text(prot4_tests),
                        test_spacing_line = 0.9,
                        test_fig =c(0, 0.5, 0.72, .99),
                        test_cex = 1.5,
                        fitAnalysis_fig = c(0,.7,.05,.5),
                        file_name = "Prot4_Biplot2D",
                        directory = directories$prot4,
                        width = 1000, height = 1000,
                        output_type = "eps")

# Biplot 3D ------------------------------------------------------------------

biplot2d3d::biplot_3d(prot4_3d,
                        ordination_method = "PCoA",
                        point_type = "point",
                        groups = factor_list$FabricGroup,
                        group_color = color_list$FabricGroup,
                        group_representation = "stars",
```

```
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot4_3d$sub3D,
                      test_text = prot4_tests$text(prot4_tests),
                      test_cex = 1.25,
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      view_zoom = 0.9)

# save animated GIF or PNG snapshot
biplot2d3d::animation(directory = directories$prot4,
                       file_name = "Prot4_Biplot3D")
```

```
[1] "Esto es la impresión integral del archivo 'Protocol_4_Shipwreck.R'"

# License notice:
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#
############################################################
# cerUB - Protocol 4 (with shipwrecks, no true outlier)
#
```

```
# 1. geochemical compositional data (CoDa) & Ordinal petrographic data (Provenance sel
# 2. centred log-ratio transformation (clr) & transformed to ranks
# 3. Extended Gower Coefficient: Euclidean distance & Relative ranking
# 4. Principal Coordinates Analysis (PCoA)
# 4. PERMANOVA & PERMDISP tests
#
############################################################


# NOTE: "Initial procedures.R" must be ran once first.


# Apply protocol 4 to identify the provenance of shipwreck samples ------------------

# project in 2d (default)

prot4_Shipwreck_2d <- apply_ordination(cleanAmphorae[!isTrueIND,], # with shipwrecks,
                                       "4", # select protocol 4
                                       exception_columns = excep_cols,
                                       variable_tags = varCode,
                                       coda_override = chemVars16,
                                       coda_transformation = "CLR")

# Simplify CoDa names for plot clearness
prot4_Shipwreck_2d <- simplify_coda_names(prot4_Shipwreck_2d)


# project in 3d (dimensions = 3)

prot4_Shipwreck_3d <- apply_ordination(cleanAmphorae[!isTrueIND,], # no shipwrecks
                                       "4", # select protocol 4
                                       exception_columns = excep_cols,
                                       variable_tags = varCode,
                                       coda_override = chemVars16,
```

```
                                        coda_transformation = "CLR",
                                        dimensions = 3)

# Simplify CoDa names for plot clearness
prot4_Shipwreck_3d <- simplify_coda_names(prot4_Shipwreck_3d)

# Test the given provenance groups --------------------------------

# Note1: Tests depend only on the distance matrix and the groups, not on the projectio
# Note2: This test batch may take several minutes depending on the size of the
# data matrix and the number of groups.
prot4_Shipwreck_tests <- test_groups(prot4_Shipwreck_2d$dist_matrix,
                                     factor_list_Shipwreck$ProvGroup)

# Biplot 2D -------------------------------------------------------------

arrows_label_adj <- rbind(c(.5,0),c(.5,1),c(.5,0),c(.5,1),c(.5,0),c(.5,1),c(.8,0),
                          c(1,.5),c(.5,0),c(1,.2),c(.5,1),c(.2,.7))
row.names(arrows_label_adj) <- c("S7","S8","S4","CaO","MgO","S11","L48",
                                 "SiO2","Ce","Nb","Th","TiO2")

# better "preview" (R UI device) version
biplot2d3d::biplot_2d(prot4_Shipwreck_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      ylim = c(-.3,.25),
                      point_type = "point",
                      groups = factor_list_Shipwreck$ProvGroup,
                      group_color = color_list_Shipwreck$ProvGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
```

```
                      subtitle = prot4_Shipwreck_2d$sub2D,
                      test_text =
                        prot4_Shipwreck_tests$text(prot4_Shipwreck_tests),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.62, .99),
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      output_type = "preview")

# better PNG version
biplot2d3d::biplot_2d(prot4_Shipwreck_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      grid_cex = 2.5,
                      ylim = c(-.3,.25),
                      point_type = "point",
                      groups = factor_list_Shipwreck$ProvGroup,
                      group_color = color_list_Shipwreck$ProvGroup,
                      group_label_cex = 1.5,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 2,
                      arrow_cex = 0.2,
                      arrow_lwd = 2.5,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot4_Shipwreck_2d$sub2D,
                      subtitle_cex = 2.5,
                      test_text =
                        prot4_Shipwreck_tests$text(prot4_Shipwreck_tests),
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      test_cex = 2,
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      file_name = "Prot4_Shipwreck_Biplot2D",
                      directory = directories$prot4_Shipwreck,
```

```
                      width = 1000, height = 1000,
                      output_type = "png")


# better EPS version
biplot2d3d::biplot_2d(prot4_Shipwreck_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      grid_cex = 2.5,
                      ylim = c(-.3,.25),
                      point_type = "point",
                      groups = factor_list_Shipwreck$ProvGroup,
                      group_color = color_list_Shipwreck$ProvGroup,
                      group_label_cex = 1.5,
                      arrow_mim_dist = .5,
                      arrow_label_cex = 1.5,
                      arrow_cex = 0.2,
                      arrow_lwd = 2.5,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      subtitle = prot4_Shipwreck_2d$sub2D,
                      subtitle_cex = 2.5,
                      test_text = prot4_Shipwreck_tests$text(prot4_Shipwreck_tests),
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      test_cex = 1.5,
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      file_name = "Prot4_Shipwreck_Biplot2D",
                      directory = directories$prot4_Shipwreck,
                      width = 1000, height = 1000,
                      output_type = "eps")


# Biplot 3D ------------------------------------------------------------
```

```
biplot2d3d::biplot_3d(prot4_Shipwreck_3d,
                      ordination_method = "PCoA",
                      point_type = "point",
                      groups = factor_list_Shipwreck$FabricGroup,
                      group_color = color_list_Shipwreck$FabricGroup,
                      group_representation = "stars",
                      star_centroid_radius = 0,
                      star_label_cex = .8,
                      arrow_min_dist = .5,
                      arrow_body_length = .025,
                      subtitle = prot4_Shipwreck_3d$sub3D,
                      test_text =
                        prot4_Shipwreck_tests$text(prot4_Shipwreck_tests),
                      test_cex = 1.25,
                      test_spacing_line = 0.9,
                      test_fig =c(0, 0.5, 0.72, .99),
                      view_zoom = 0.9)

# save animated GIF or PNG snapshot
biplot2d3d::animation(directory = directories$prot4_Shipwreck,
                       file_name = "Prot4_Shipwreck_Biplot3D")
```

```
[1] "Esto es la impresión integral del archivo 'Interpreting_biplots.R'"

# License notice:
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#
############################################################
# cerUB - Appendix on interpreting biplots
############################################################


# NOTE: "Initial procedures.R", "Protocol_1.R", and "Protocol_2.R" must be ran once fi

# Protocol 1 example----

# Let's recover protocol 1 override for variable label positions
arrows_label_adj <- rbind( c(.5, -.5), c(1, .5), c(1.2, 1.2), c(1.2, .4),
                           c(.8, .5),
                           c(0, 0), c(-.2, 1), c(.5, 1.2), c(-.5, .5),
                           c(-.2, .5), c(0, .5), c(0, 0))
row.names(arrows_label_adj) <- c("Fe2O3", "Al2O3", "SiO2", "TiO2",
                                 "MgO",
                                 "Th", "Nb", "Cr", "Ce",
                                 "Ga", "Zn", "Y")

# Protocol 1, representing and testing chemical reference groups
biplot_2d(prot1,
          groups = factor_list$ChemGroup,
          group_color = color_list$ChemGroup,
          group_label_cex = 0.6,
          invert_coordinates = c(TRUE, TRUE),
          arrow_label_cex = 0.7,
          test_text = prot1_tests$text(prot1_tests),
          test_cex = 0.8,
          test_fig = c(0, 0.5, 0.65, .99),
```

```
          output_type = "preview")


# Interpret Protocol 1 in terms of CaO

# Create factor variable containing the classification (5 categories)
CaO_level <- cut(cleanAmphorae$CaO[!isShipwreck], 5)

# Select 5 colours from the 'topo.colors' palette
CaO_level_colors <- topo.colors(nlevels(CaO_level))

# Test the classification
prot1_tests_CaO <- test_groups(prot1$dist_matrix, CaO_level)

# This is for highlighting CaO arrow
arrow_colors <- rep("darkorange", nrow(prot1$loadings))
arrow_colors[row.names(prot1$loadings) == "CaO"] <- "red"

# Protocol 1, grouping by level of CaO content
biplot_2d(prot1,
          groups = CaO_level,
          group_color = CaO_level_colors,
          group_star_cex = 0,
          group_label_cex = 0,
          show_group_legend = TRUE,
          group_legend_title = "CaO",
          group_legend_title_pos = c(0.5,0.9),
          group_legend_text_cex = 0.8,
          group_legend_fig = c(0.7,0.99,0.68,0.95),
          invert_coordinates = c(TRUE, TRUE),
          arrow_label_cex = 0.8,
          arrow_fig = c(.6,.95,0,.35),
          arrow_label_adj_override = arrows_label_adj,
          arrow_color = arrow_colors,
```

```
          test_text = prot1_tests_CaO$text(prot1_tests_CaO),
          test_cex = 0.8,
          test_fig = c(0, 0.5, 0.65, .99),
          output_type = "preview")

#----

# Protocol 2 examples----

# Let us recover protocol 2a override for variable label positions
arrows_label_adj <- rbind(c(.5,.8),c(.5,1),c(.5,1),c(.5,0),c(.5,1),
                          c(.5,0),c(0,.5))
row.names(arrows_label_adj) <- c("L48","L24","L5","L36","S7",
                                 "S8","S11")

# This will help us select different arrow colours
isDisplayed <-
  row.names(prot2a_2d$loadings) %in% row.names(
    filter_arrows(prot2a_2d$loadings, min_dist = 0.5))

# protocol 2a, representing and testing fabric groups
biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
                      point_type = "point",
                      groups = factor_list$FabricGroup,
                      group_color = color_list$FabricGroup,
                      group_label_cex = 0.6,
                      arrow_mim_dist = 0.5,
                      arrow_label_cex = 0.6,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
```

```
                      subtitle = prot2a_2d$sub2D,
                      test_text = prot2a_tests$text(prot2a_tests),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      fitAnalysis_fig = c(0,.7,.05,.5),
                      output_type = "preview")

# Interpret Protocol 2a in terms of INCLUS_ORIENT

# You may want to assure that the true categories are corectly represented:
cleanAmphorae <- order_petro(cleanAmphorae)

levels(cleanAmphorae$INCLUS_ORIENT[!isShipwreck])

# Let us declare this factor separately as an object for clearness
I2 <- cleanAmphorae$INCLUS_ORIENT[!isShipwreck]

# Select colours from the 'topo.colors' palette
I2_colors <- topo.colors(nlevels(I2))

# Test the classification
prot1_tests_I2 <- test_groups(prot2a_2d$dist_matrix, I2)

# This is for highlighting CaO arrow
arrow_colors <- rep("darkorange", nrow(prot2a_2d$loadings))
arrow_colors[row.names(prot2a_2d$loadings) == "I2"] <- "red"
# filter arrows colours, since not all variables are displayed
arrow_colors <- arrow_colors[isDisplayed]

# Protocol 2a, grouping by INCLUS_ORIENT
biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
```

```
                        ylim = c(-.31,.35),
                        groups = I2,
                        group_color = I2_colors,
                        group_star_cex = 0,
                        group_label_cex = 0,
                        show_group_legend = TRUE,
                        group_legend_title = "INCLUS_ORIENT",
                        group_legend_title_pos = c(0.5,0.9),
                        group_legend_text_cex = 0.8,
                        group_legend_fig = c(0.6,0.99,0.68,0.95),
                        arrow_mim_dist = .5,
                        arrow_label_cex = 0.8,
                        arrow_fig = c(.6,.95,0,.35),
                        arrow_label_adj_override = arrows_label_adj,
                        arrow_color = arrow_colors,
                        subtitle = prot2a_2d$sub2D,
                        test_text = prot1_tests_I2$text(prot1_tests_I2),
                        test_cex = 0.8,
                        test_fig = c(0, 0.5, 0.65, .99),
                        output_type = "preview")

# Interpret Protocol 2a in terms of COAR_R_CHERT

# Let us declare this factor separately as an object for clearness
L33 <- cleanAmphorae$COAR_R_CHERT[!isShipwreck]

# Select colours from the 'topo.colors' palette
L33_colors <- topo.colors(nlevels(L33))

# Test the classification
prot1_tests_L33 <- test_groups(prot2a_2d$dist_matrix, L33)

# This is for highlighting CaO arrow
arrow_colors <- rep("darkorange", nrow(prot2a_2d$loadings))
```

```
arrow_colors[row.names(prot2a_2d$loadings) == "L33"] <- "red"
# filter arrows colours, since not all variables are displayed
arrow_colors <- arrow_colors[isDisplayed]

# Protocol 2a, grouping by COAR_R_CHERT
biplot2d3d::biplot_2d(prot2a_2d,
                      ordination_method = "PCoA",
                      invert_coordinates = c (TRUE,TRUE),
                      xlim = c(-.26,.35),
                      ylim = c(-.31,.35),
                      groups = L33,
                      group_color = L33_colors,
                      group_star_cex = 0,
                      group_label_cex = 0,
                      show_group_legend = TRUE,
                      group_legend_title = "COAR_R_CHERT",
                      group_legend_title_pos = c(0.5,0.9),
                      group_legend_text_cex = 0.8,
                      group_legend_fig = c(0.6,0.99,0.68,0.95),
                      arrow_mim_dist = .5,
                      arrow_label_cex = 0.8,
                      arrow_fig = c(.6,.95,0,.35),
                      arrow_label_adj_override = arrows_label_adj,
                      arrow_color = arrow_colors,
                      subtitle = prot2a_2d$sub2D,
                      test_text = prot1_tests_L33$text(prot1_tests_L33),
                      test_cex = 0.8,
                      test_fig = c(0, 0.5, 0.65, .99),
                      output_type = "preview")
```

A.2 Modelización y simulación computacional

5.3.5. A.2.1 Apéndices de 'Land Use Patterns in Central Asia. Step 1: The Musical Chairs Model' { - }

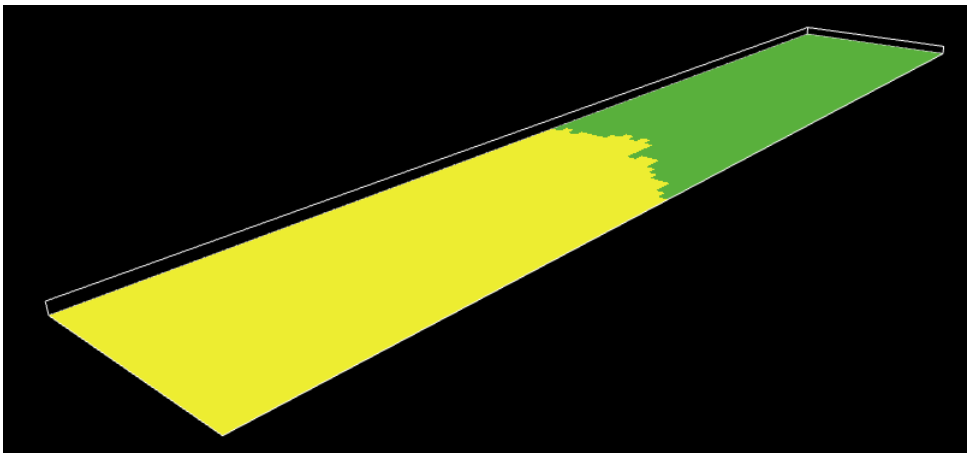
El apéndice de Angourakis et al. (2014) es:

- Appendix: The Musical Chairs model (ODD)

SimulPast Project - Case Study 5

Oasis construction in Central Asia: *The interaction of herding and farming land uses in oases* of Central Asia. Step 1: Competition for land use

The Musical Chairs Model (ODD Protocol)



Andreas Angourakis (a), Bernardo Rondelli (b), Sebastian Stride (a), Xavier Rubio (c), Andrea L. Balbo (b), Verònica Martinez (a), Marco Madella (b, d) and Josep M. Gurt (a)

(a) ERAAUB, Department of Prehistory, Ancient History and Archaeology, University of Barcelona, C/Montalegre, 6, 08001 Barcelona, Spain.

(b) Complexity and Socio-Ecological dynamics (CaSEs), Istitució Milà i Fontanals, Spanish National Research Council (IMF-CSIC), C/Egipciaques 15, 08001 Barcelona, Spain.

(c) Barcelona Supercomputing Center (BSC), Nexus II Building, C/ Jordi Girona, 29, 08034 Barcelona, Spain.

(d) Catalan Institute of Research and Advanced Studies (ICREA).

Overview, **D**esign concepts and **D**etails protocol

Overview

Purpose

This Agent-Based model intends to explore the conditions for the emergence and change of land use patterns in Central Asian oases and similar contexts. Land use pattern is conceptualized as the proportion between the area used for mobile livestock breeding (herding) and sedentary agriculture (farming), the main forms of livelihood from the Neolithic to the Industrial Revolution. We assume that these different forms of land use interact in recurrent competitive situations, given that the land useful for both activities at the same time is limited and there is a pressure to increase both land uses, due to demographic and/or economic growth. As our intention is to create a start point for developing new theories on oases construction in Central Asia, the Musical Chairs model do not represent explicitly the processes underlying land use change, but keep them in a "black box" to be further investigated with more complex models. Furthermore, we have also kept its variables relatively simple, so they hardly can be validated by real data at this stage (e.g. it does not generates spatial patterns).

Entities, state variables and scales

The world of the model is a set of scale-free land units (i.e. patches of land with arbitrary size). It represents an area next to a river, covering both the river banks and the surrounding terrain (i.e. alluvial plain or cone), which are assumed to be able to accommodate either farmlands or pastures.

Agents are the land use variants that can be assigned to land use units. They are differentiated by main land use (`farming`, `herding`), `intensity` and `independence`. Their main (i.e. most extended) land use is represented as the agents' class, while `intensity` and `independence` are agent-level variables (i.e. traits), fixed during the lifetime of the agent. `Intensity` stands for the relative amount of productive factors involved in a land use variant. `Independence` expresses how much a land use variant does not depend on variants with different main land use (e.g. by sharing productive factors). `Independence` is a value between 0 and 1, while `intensity` ranges between 0 and an arbitrary maximum. The maximum for intensity is class-specific and the difference between classes is defined as the parameter `herding_relative_maximum_intensity` (e.g. if its value is 5, then herding is able to achieve five times more intensity than farming). To consider agents to be land use variants instead of people or groups may be a less intuitive and straightforward choice, but allows us to account for all the variety of cases in a rather simpler approach. For instance, competition between land uses could be even a dilemma internal to an individual, with each agent representing a possible individual choice about how to use the land in a specific context.

`Herding` and `farming` land uses are global numerical variables, accounted as land units out of a total given by the size of the world considered in the model. They are calculated at every time step from the population of `farming agents` and they are eventually updated after competition is resolved. Differently to most Agent-Based models, agents are situated randomly in the artificial world, and all processes are independent of those positions. All parameters and variables are displayed in Fig. 1.

Process overview and scheduling

The scheduling of the model consists in a four-step cycle (Fig. 2): the expansion of both land uses (`farming_expansion`, `herding_expansion`), the re-structuring of land use pattern (`update_land_use`) and the checking and resolution of competitive situations between farming and herding, reiterated for each herding agent without access to a land unit (`check_competition`).
First, in the `farming_expansion` and `herding_expansion` procedures, there are four sub-procedures affecting the respective populations of agents:

1. *Intrinsic growth*: the agents are duplicated, depending on a certain probability per time step (`farming_intrinsic_growth_rate`, `herding_intrinsic_growth_rate`);
2. *Extrinsic growth*: agents with random traits are created, up to the number of land units not occupied by the same class, depending on a certain probability per time step (`farming_extrinsic_growth_rate` and `herding_extrinsic_growth_rate`);
3. *Fit-to-maximum exclusion*: both populations are checked to fit within the maximum of the artificial world, while any excess of agents is excluded of the simulation following a random order;
4. *Density-dependent exclusion*: recently created agents of each population are deleted with a probability proportional to the land still available for their type.

Moreover, new farming agents perform yet another procedure, *volition-opportunity exclusion*, in which they test their particular `independence` against the proportion of land currently used by herding (`herding`), as a proxy of the probability of having its extension curbed by their dependency to herding (e.g. if farmers have interest also in the welfare of herds): if the former is lower than the latter, the farming land use variant will be deleted. All excluded agents during these procedures are represented in the output variables `farming_deterrence` and `herding_deterrence` and account for the numerous phenomena that may imply having to discard potential farms or herds due to limiting conditions (e.g. increasing mortality rates, emigrant fluxes moving to any adjacent territory, households changing their livelihood).
The `update land use` procedure assigns the values of the realization of the two land uses, `farming` and `herding`. `Farming` land use is equal to the number of farming agents present in the territory, whilst the amount of land available for herding (i.e. pastures) is given by the difference between the total number of land units and the ones that are used for farming.
The `check_competition` sub-cycle is the reiteration of the resolution of competitive situations between herding and farming (`resolve_competition`), either until all land units required by herding is taken and reconverted into pastures, or the population of herding agents is reduced to the maximum sustained by the current land use pattern. Therefore, its core process (`resolve_competition`) only is activated when there are more herding agents than herding land units (i.e. if there are fewer agents than land units, there is no competitive situation).

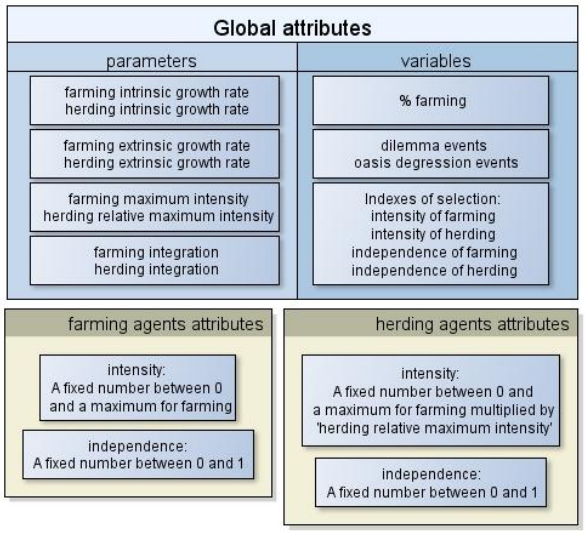


Fig. 1: Global and agent attributes

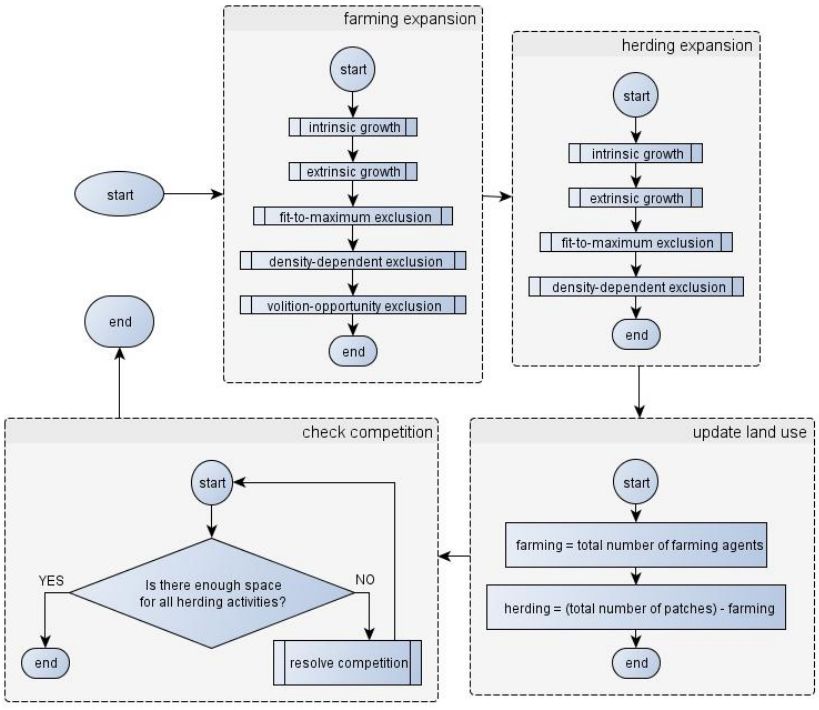


Fig. 2: Flowchart depicting the scheduling and submodels of the Musical Chairs Model

Design concepts

Basic principles

The starting point of the Musical Chairs model is that sedentary agriculture and pastoralism are two livelihoods that entail qualitatively different land uses, and that these land uses are mutually exclusive during a part of the year. Given the assumptions that the demand of land use grows and that the land useful for both activities remains constant, we postulate that there is a recurrent competitive situation between farming and herding. This model presents a solution for such competition, according to which the land use pattern will depend on the intensity, independence and current extension of both classes of land use. Because there are no pay-offs for agents that choose to exit the system (i.e. land use variants that are discarded), the Musical Chairs model cannot be classified as a game (i.e. strategic agents), but simply as a variation-selection process (i.e. rule-based agents). Therefore, its interest relies only in measuring the conditions in which this competition will favour one or another land use.

Emergence

The border between farmlands and pastures is constrained by the possibility of cultivation, imposed in the model by the total number of land units; i.e. farming cannot be extended beyond the world of the model. However, the factual border emerges from the competition between variants of land use characterized by both farming and herding.

Adaptation

In this model agents have a rough capacity for adapting their behaviour by choosing to exit the system or to risk losing their land unit in a dilemma event. However this is actually a selective process from the oasis perspective, since exiting the territory equals discarding a land use variant. True adaptation from the agent perspective will be only possible with the assumption of self-regulating agents (i.e. agents deciding to modify their own characteristics in order to better cope with stress) or considering various interconnected territories simultaneously, instead of assuming regional conditions to always present mid-level opportunity and risk. However, introducing these aspects will not change the results locally, which strictly depend on the external selective factors (i.e. parameters).

Objectives/Fitness

The main goal of all agents is to "survive" as agents, i.e. to acquire and keep the one land unit they need in order to exist as land use variants. In order to reflect the rationale of real decision-making regarding land use, agents' objectives are summarized as the following rules:

farming
1. Choose the most promising territory, given that the options outside the system are assumed to sum up a mid-level opportunity (i.e. 50% of their land units on average may be freely used for farming);
2. Choose the territory with the least probability of suffering a dilemma event, given that the options outside the system are assumed to sum up a mid-level risk of conflict (i.e. 50% of their land units on average are already used for herding);
3. Once settled in the territory, never consider leaving (i.e. farming investments are not movable).

herding
1. Choose the most promising territory, given that the options outside the system are assumed to sum up a mid-level opportunity (i.e. 50% of their land units on average may be freely used for herding);

2. Choose the territory with the least probability of suffering a dilemma event, given that the options outside the system are assumed to sum up a mid-level risk of generating a dilemma event (i.e. 50% of their land units on average are already used for farming);
3. Once established a territory as a herding route, consider changing it whenever pastures are scarce (i.e. the territory becomes too dominated by either farming or herding)

Agents' fitness is considered through a clear-cut distinction between agents that exists and those that were excluded. Evolution can occur on both populations, through the selection of the traits `intensity` and `independence`. Although the logic of the model previously defines that the values of these traits have a positive relationship with agents' fitness on average, the presence and strength of selection can vary significantly, depending on the conditions given by the parameters.

Learning

There is no learning capacity for agents in the Musical Chairs model. The traits characterizing individual agents, `intensity` and `independence`, are assumed to be not associated with individual learning and intentions. This assumption was made in order to define agents strictly as land use variants, themselves related to an undetermined set of productive factors and decision-makers, and to explore how suboptimal land use variants would survive in different circumstances.

Prediction

As a proxy of prediction, farming agents estimate the risk of participating in a dilemma event (i.e. volition-opportunity exclusion) and herding agents can access the probability of successfully displacing an farming agent (i.e. "1 – `incentive_to_relinquish`") before actually initiating a dilemma event.

Sensing

All agents can sense the quantities of land units both used and potentially usable by their class (i.e. `farming`, `herding`), in order to evaluate the general *opportunity* and *risk* of the territory.

Interaction

Interaction only occurs when a herding agent founds itself without a land unit, and decides to initiate a dilemma event. The interaction arises between this agent and a farming agent (i.e. the two *unlucky* agents) and, if it is the case, their respective supporters. The outcome of this interaction is the deletion of one or another *unlucky* agent, while supporters suffer no consequences.

Stochasticity

Stochasticity is introduced while duplicating agents during the `farming_expansion` and `herding_expansion` processes, while assigning values for `intensity` and `independence` to all agents introduced at initialization and during simulation (i.e. extrinsic growth), while choosing agents for deletion due to the overgrowth of land use demand (i.e. fit-to-maximum and density-dependent exclusions), while choosing agents for engaging in competition and finally while testing the `ratio_of_intensities` as a probability of success of herding during dilemma events.

Collectives

In the Musical Chairs model, the only collectives that affect the outcome are the rather diffuse entities implied in the parameters `farming_integration` and `herding_integration`. These, however, do not behave as agents own their own, but as superposed clusters of the support network existing between agents of the same kind. Therefore, their action will depend directly on the ones taken by the "unlucky" agents summoning them.

Observation

Results of the Musical Chairs can be analysed through four sets of variables, included in Table 2:

1. The proportions of the territory that are involved in each land use, found in `farming`, `herding` or both. If the system is saturated with agents, a straight-forward way to assess land use pattern is a percentage. A more intuitive visualization of this proportion, using green (`farming`) and yellow (`herding`) patches, is available at the model´s interface (Fig. 3, upper-right corner).
2. The variables counting the number of agents that come "in" or "out" the territory during a time step (`farming_growth`, `farming_deterrence`, `herding_growth`, `herding_deterrence`) indicate if a state of the system is receiving or expelling agents of a given class. The difference between growth and deterrence (`farming_balance`, `herding_balance`) point to the existence of equilibrium and its stability.
3. The variables measuring the occurrence of dilemma and oasis degression events, are indicative of the level of potential and actual land use change in the territory during a particular time step.
4. The variables that infer the existence and strength of selection of agents' traits (`mean_fint`, `mean_find`, `mean_hint`, `mean_hind`), by calculating the mean between four groups, weighted by the subpopulation they represent (e.g. if all herding agents have an intensity superior to 75% of the maximum value, then `mean_hint` will be between 3 and 4).

NetLogo's interface offers a graphical visualization of these variables (FIG 3).

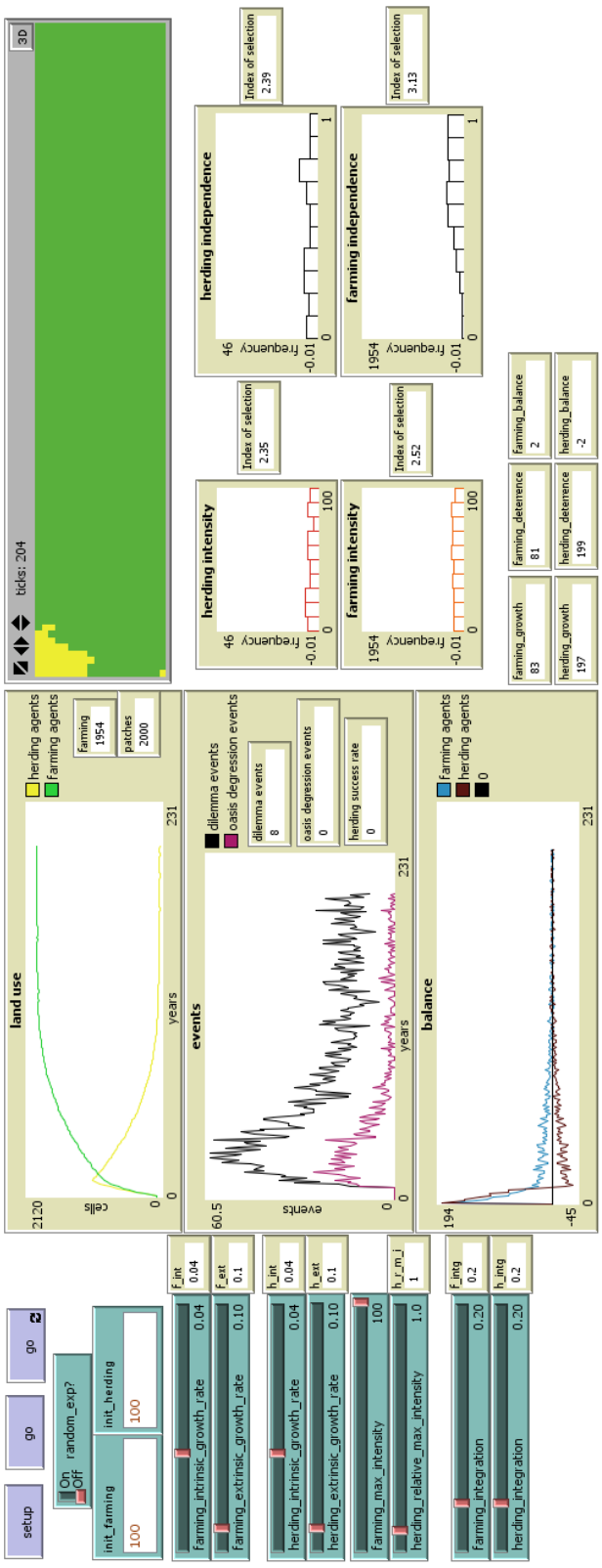


FIG. 3: Snapshot depicting the model´s interface in *NetLogo*

Details

Initialization

In order to be simulated, the model is initialized as follows. First, the world size is defined using as parameter the maximum values for two spatial dimensions (`max-pxcor`, `max-pycor`). Then, both populations of agents are generated according to the quantities specified by `init_farming` and `init_herding`. Farming agents will be randomly-assigned a value of `intensity` ranging from 0 to `max_farming_intensity`, and herding agents will have an also randomly-assigned `intensity`, but ranging from 0 to the product of `max_farming_intensity` and `herding_relative_max_intensity`. However, if the simulation is a randomized experiment, initial populations will be randomly-chosen integers between 0 and the respective values of `init_farming` and `init_herding`, and the value of `herding_relative_max_intensity` will be replaced by a random rational number between `herding_relative_max_intensity` and its reciprocal (i.e. `herding_relative_max_intensity`$^{-1}$), following a skewed distribution with mean 1. Finally, each agent is randomly-assigned a particular value for the variable `independence`, varying between 0 and 1. All random numbers are chosen from a uniform probability distribution, unless it is stated otherwise.

Input

The model does not use input data to represent time-varying processes.

Submodels

All submodels of the Musical Chairs model represent different processes through which agents are either generated or sorted out of the system. They are here presented following their order in the model's schedule.

Intrinsic growth

The *intrinsic growth* submodel includes the assumption that (1) the demand of land use grows in absence of (exogenous) limiting factors, and (2) it does this following a constant rate. Moreover, (3) growth rates associated with farming and herding land uses were assumed to be equal and (4) fixed at 0.04, a realistic overall value for historical periods preceding the industrial era (note that this value does not represent the actual grow rate, but the maximum grow rate).

This submodel generates pseudo-random numbers between 0 and 1, out of a uniform distribution, for each agent of a population; test it against the fixed rate of growth for the respective population (`farming_intrinsic_growth_rate`, `herding_intrinsic_growth_rate`) and duplicates all agents with a number equal or smaller than this rate. This procedure imitates, in an Agent-based fashion, the well-known model of exponential growth.

Extrinsic growth

The basic assumption behind the *extrinsic growth* submodel is that there is a set of agents of a given type willing to immigrate to the model's territory, if the latter presents them an opportunity. These two sets of agents (one for each type) represent indistinctly the potential fluxes of

productive factors coming from neighbouring territories not represented in the model. Moreover, the source is assumed to be *unlimited* (i.e. incoming agents could potentially occupy all the territory in a single time-step) and *inexhaustible* (i.e. there will always be potential agents to enter the territory, whatever the volume of past flows).

This submodel first calculates the attractiveness of the territory, in terms of how many land units can further be occupied by a given class of agent: whereas for farming it is equal to `herding`, for herding agents it is the difference between the total of land units and the number of herding agents that were at the territory in the previous time-step. Subsequently, this value is weighted by the externally-given and constant proportion of potential immigrants that will actually enter the territory (`farming_extrinsic_growth_rate`, `herding_extrinsic_growth_rate`) and then rounded. Finally, agents of the respective class are generated up to the resulting integer.

Fit-to-maximum exclusion

This submodel checks that the overall number of agents, including those recently created through intrinsic and extrinsic growth, fits the maximum given by the total amount of land units in the territory. Each new agent will check this condition in a random order and, if not complied, it will be excluded. This mechanism –according to which there cannot be more agents of each class than land units— is a necessary implication of the definition of agents and spatial units in this model: agents are not representation of specific individuals or groups of individuals, but arbitrary units of land use.

Density-dependent exclusion

Similarly to the *Verhulst-Pearl* or *logistic* equation, the density-dependent submodel performs a correction on exponential growth and it penalizes the occupation of peripheral land units. This submodel is based in the assumption that the more extended is a land use is a territory, the lesser the incentives to further extend it.

The density-dependent exclusion sub model is compounded by a single procedure, in which the density of a class of land use (i.e. the number of agents of that class divided by the total amount of land units) is tested against a pseudo-random number between 0 and 1, out of a uniform distribution, for each new agent of that specialization following a random order. Therefore, the density of a population of agents is treated as a proxy of the probability that a specific agent will exit the system.

Volition-opportunity exclusion

This submodel applies only to farming agents, and it describes how the current landscape land use pattern can stimulate or restrain the expansion of farming. It involves two assumptions: (1) that dilemma events are undesirable for decision-makers involved in farming, and (2) that such decision-makers decide to settle new farmlands before the competitive season (i.e. the arrival of herds). The latter imply that the foundation of new farmlands will be done with poor information on if and by which herders a given area will be claimed as pastures. In this sense, the only source of information available is the landscape itself, perceived as the proportion of `herding` land units against the total of land units. On the other hand, this information will be perceived as more or less relevant, depending on how much this farming land use variant depends on herding: if it is completely independent, the decision-makers involved will always press for the expansion of farming and so the new farming agent will always survive this filter. Accordingly to this submodel, each new farming agent will test this perception against its value of

`independence` and, if it is the case that the latter is lower than the former, the agent will exit the system.

Competitive exclusion (`resolve_competition`)

This submodel accounts for the events occurring during the competitive season, in which herders come to graze their animals inside the same territory where agricultural settlements exist. It presents an answer to the problem of land use competition between these two activities during this season. Two assumptions are needed for this submodel: (1) as it was also the case for the *fit-to-maximum* exclusion, agent's requirements of one land parcel is by definition not flexible, hence not compressible (i.e. the number of agents cannot surpass the number of land units); (2) contrasting with farming agents, herding agents will have access both the extent of farming land use and the conditions of any forthcoming competition (i.e. `ratio_of_intensities`), because all agents will actually be there to be observed.

This submodel is called recursively, whenever the number of herding agents is greater than the current value of `herding` land units. Its specific procedures are the following:

1. Two agents of each kind are randomly chosen to be the ones driven into competition (the *unlucky*);
2. Particular helpers are randomly-chosen among the respective populations, according to the predefined ratios of connectivity (`farming_integration` and `herding_integration`);
3. The overall intensity of each party is summed up as `farming_intensity` and `herding_intensity`;
4. The relative ratio between the intensity of the herder party and that of its opponents (`ratio_of_intensities`, varying between 0.0 and 1.0) is calculated;

$$ratio\_of\_intesities = \frac{herding\_intensity}{herding\_intensity + farming\_intensity}$$

5. The index of opportunity regarding the amount of land units that could be gained for herding by transforming farmland into pasture (`index_of_opportunity`, varying between 0.0 and 1.0) is calculated;

$$index\_of\_opportunity = \frac{farming}{"total\ number\ of\ land\ use\ units"}$$

6. The incentive that the unlucky herding agent has for giving up the parcel and exiting the world (`incentives_to_relinquish`, varying between 0.0 and 1.0) is calculated;

$$incentives\_to\_relinquish = 1 - ratio\_of\_intensities * index\_of\_opportunity$$

7. The value of `incentives_to_relinquish` is tested against the `independence` of the unlucky herding agent:
    a. If `incentives_to_relinquish` is greater than `independence`, the unlucky herding agent will be excluded from the simulation.
    b. When the opposite is true, the unlucky herding agent will produce a dilemma event by pressing to transform a randomly-chosen farming land unit. Its

realization is given by testing the `ratio_of_intensities` as a probability of herding success (i.e. against a random number between 0 and 1):

i. If `ratio_of_intensities` exceeds this random number, the unlucky herding agent will then acquire the land unit of the unlucky farming agent, excluding this agent from the system and transforming one unit of `farming` into one of `herding`;
ii. If `ratio_of_ intensities` is lower than this random number, the unlucky herding agent is the one to be excluded and the land use pattern remains unchanged.

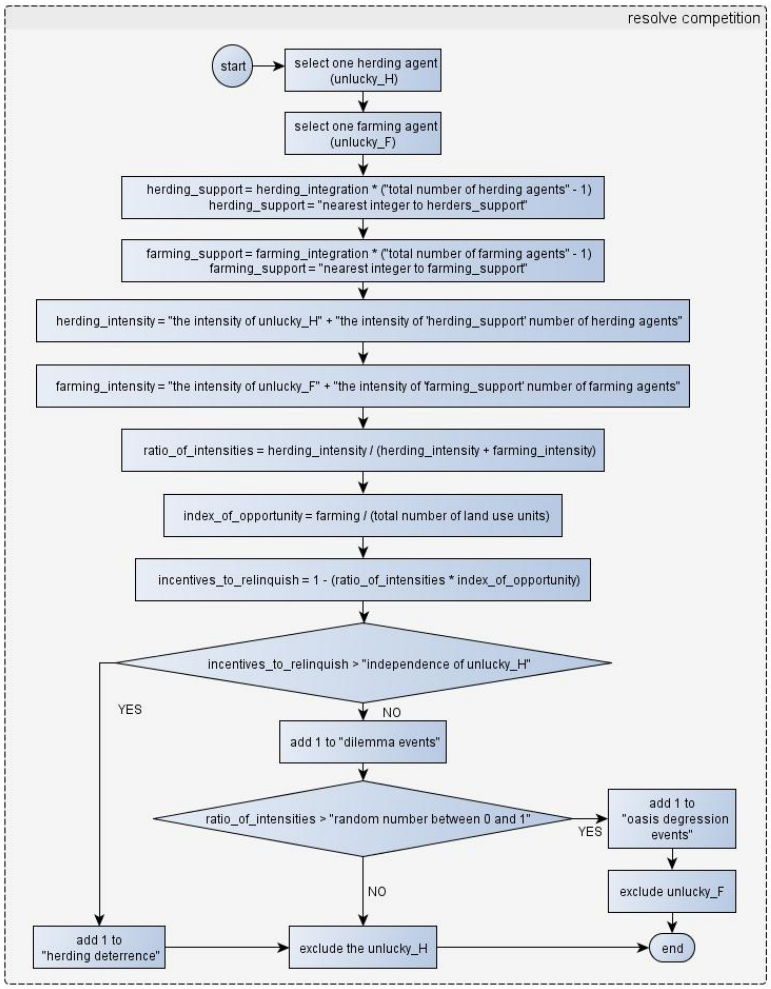


FIG. 4: Flowchart depicting the Competitive exclusion submodel (`resolve_competition`)

Table 1

Parameters

Name	*Interpretation*
Number of patches	The number of land units available in the territory.
`init_farming`	The initial number of farming land units.
`init_herding`	The initial number of herding land units.
`farming_intrinsic_growth_rate`	The probability that a farming land use variant is duplicated in another land unit during a cycle, given only its own existence.
`herding_intrinsic_growth_rate`	The probability that a herding land use variant duplicated in another land unit during a cycle, given only its own existence.
`farming_extrinsic_growth_rate`	The proportion of farming land use variants out of the total that could be further sustained in the territory, which are pressed by exogenous factors, independently of local variants.
`herding_extrinsic_growth_rate`	The proportion of herding land use variants out of the total that could be further sustained in the territory, which are pressed by exogenous factors, independently of local variants.
`farming_max_intensity`	The maximum intensity that a land use variant can have, if it is characterized by farming.
`herding_relative_max_intensity`	The ratio between the maximum value of intensity that herding land use variants can have and the one reachable by the ones characterized by farming.
`farming_integration`	The proportion of farming land use variants that are connected to a single farming land use variant.
`herding_integration`	The proportion of herding land use variants that are connected to a single herding land use variant.

Table 2

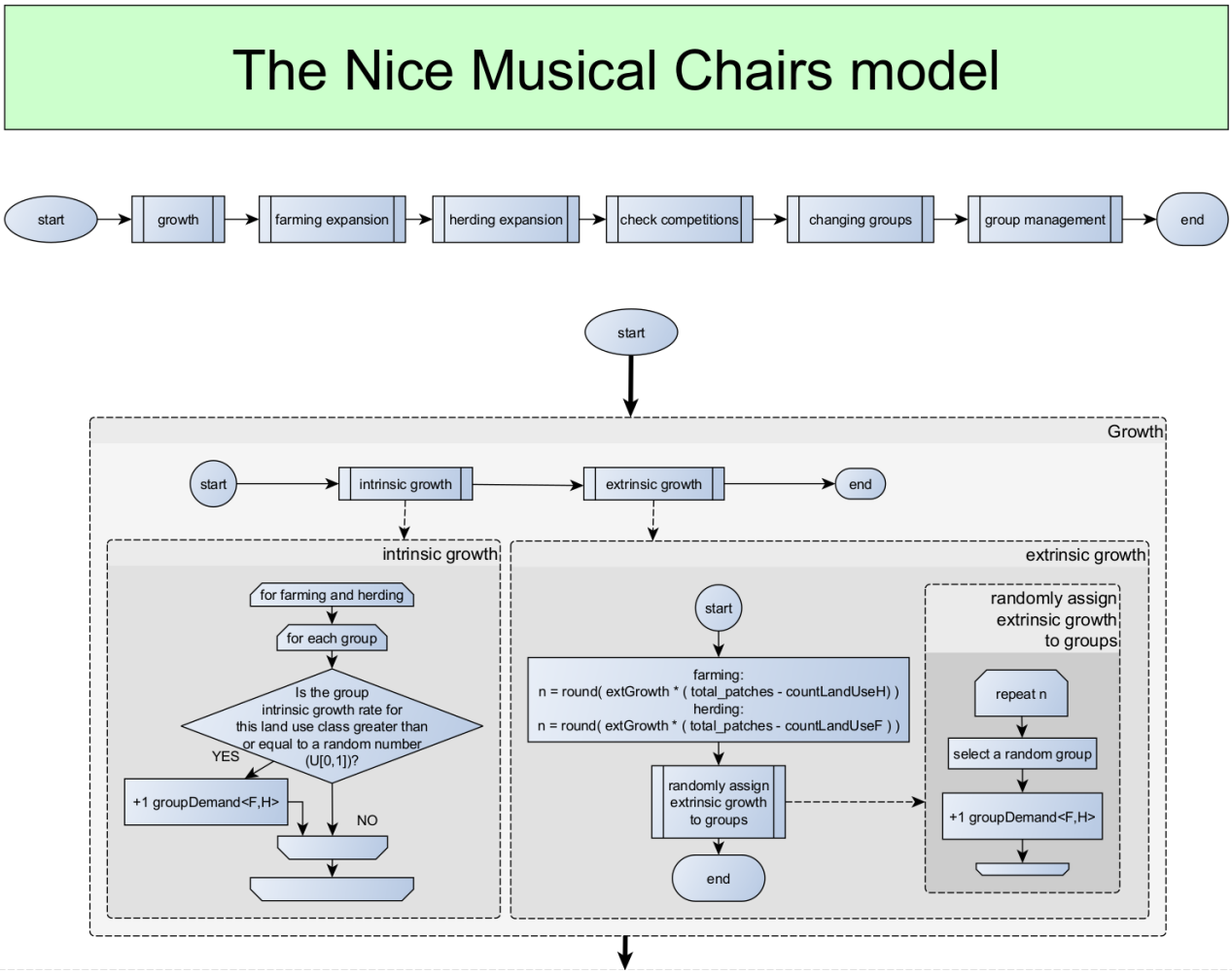
Variables

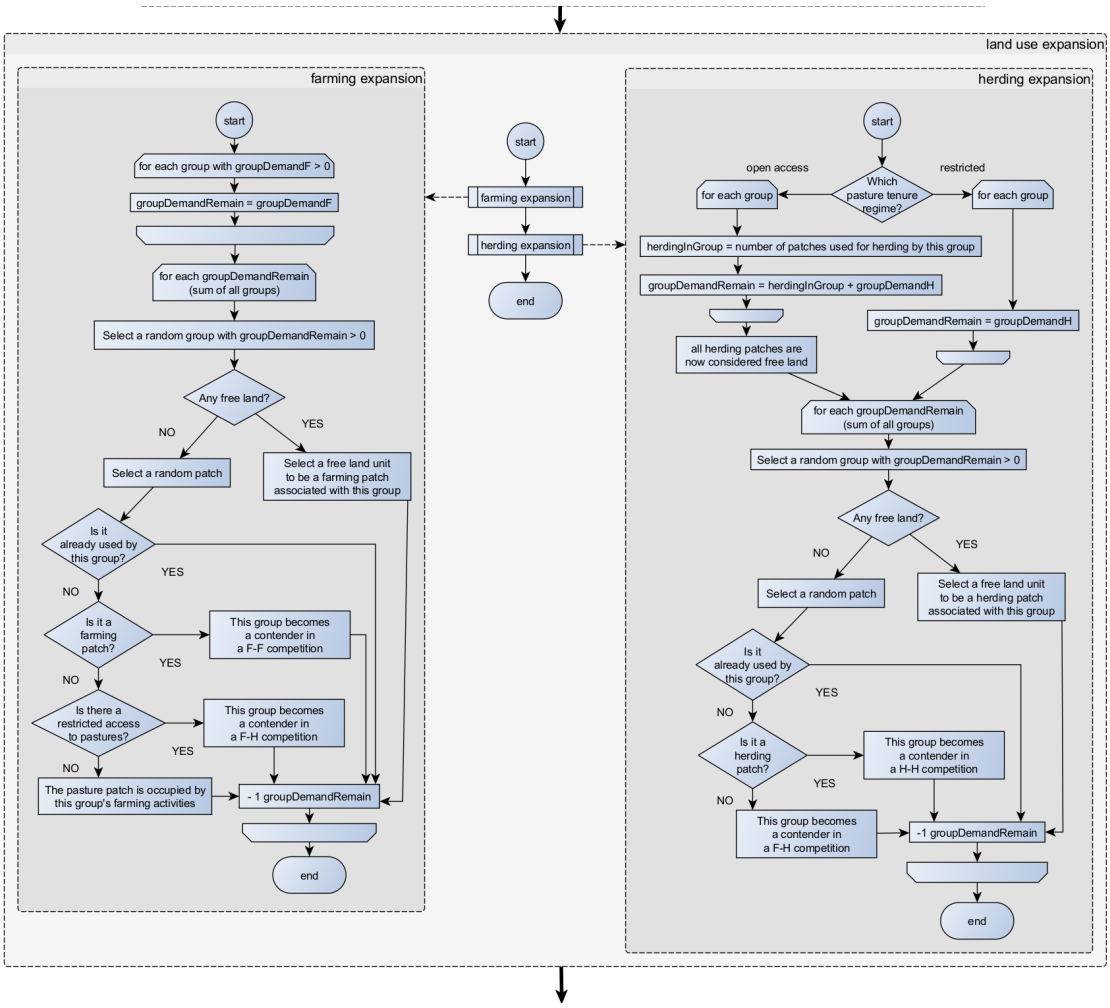
Name	*Interpretation*
`farming`	The amount of land units dominated by farming land use.
`herding`	The amount of land units dominated by herding land use.
`dilemma_events`	The number of dilemma events per time step.
`oasis_degression_events`	The number of oasis degression events per time step.
`ratio_of_intensities`	The ratio between the intensity of the *unlucky* herding land use variant and related variants, and the one of the *unlucky* farming land use variant and related variants.
`index_of_opportunity`	The ratio between the land units dominated by farming and the total number of land units in the territory; it is a measure of the potential return of turning farmlands into pastures.
`incentive_to_relinquish`	A ratio-type measure of the incentives to relinquish one herding land use variant, in opposition to the incentives to turn one land unit to herding.
`herding_success_ratio`	The proportion of dilemma events that became oasis degression events.
`independence`	The probability that the specific land use variant will be enforced at the expense of one belonging to the alternative class.
`intensity`	The capacity of a specific land use variant to enforce itself when competing with variants belonging to the alternative class.
`farming_growth`, `herding_growth`	The number of new land use variants that can occur (i.e. land use demand) during a time step.
`farming_deterrence`, `herding_deterrence`	The number of new land use variants that where discarded (i.e. frustrated land use demand) during a time step.
`farming_balance`, `herding_balance`	The difference between growth and deterrence of a land use class. Indicates the net change of a land use.
Indexes of selection of agents' traits (`intensity`, `independence`) per agent class (`mean_fint`, `mean_find`, `mean_hint`, `mean_hind`)	Indexes that identify the existence and strength of the selection on an agents' trait, representing the mean relative frequency of this trait's values separated in four groups, depending on proportions of the maximum value for this trait ([1] 0-25%, [2] 25-50%, [3] 5-75% and [4] 75-100%); if values among agents are randomly distributed (i.e. this trait is not selected), the value of the index will be approximately "2.5".

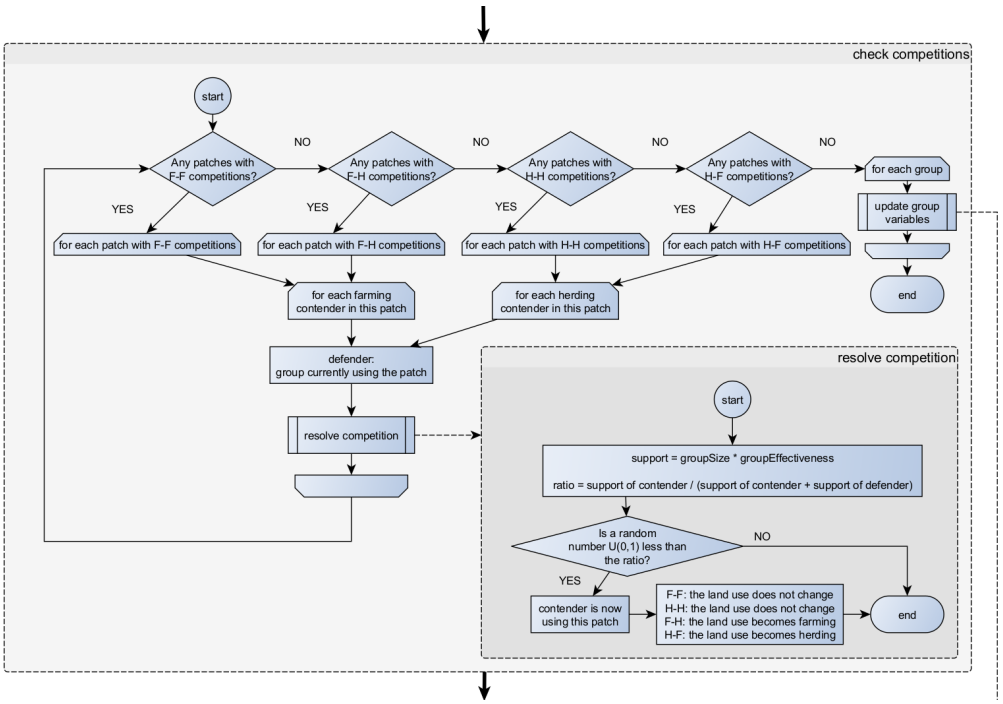
5.3.6. A.2.2 Apéndices de 'The Nice Musical Chairs Model: Exploring the Role of Competition and Cooperation Between Farming and Herding in the Formation of Land Use Patterns in Arid Afro-Eurasia' { - }

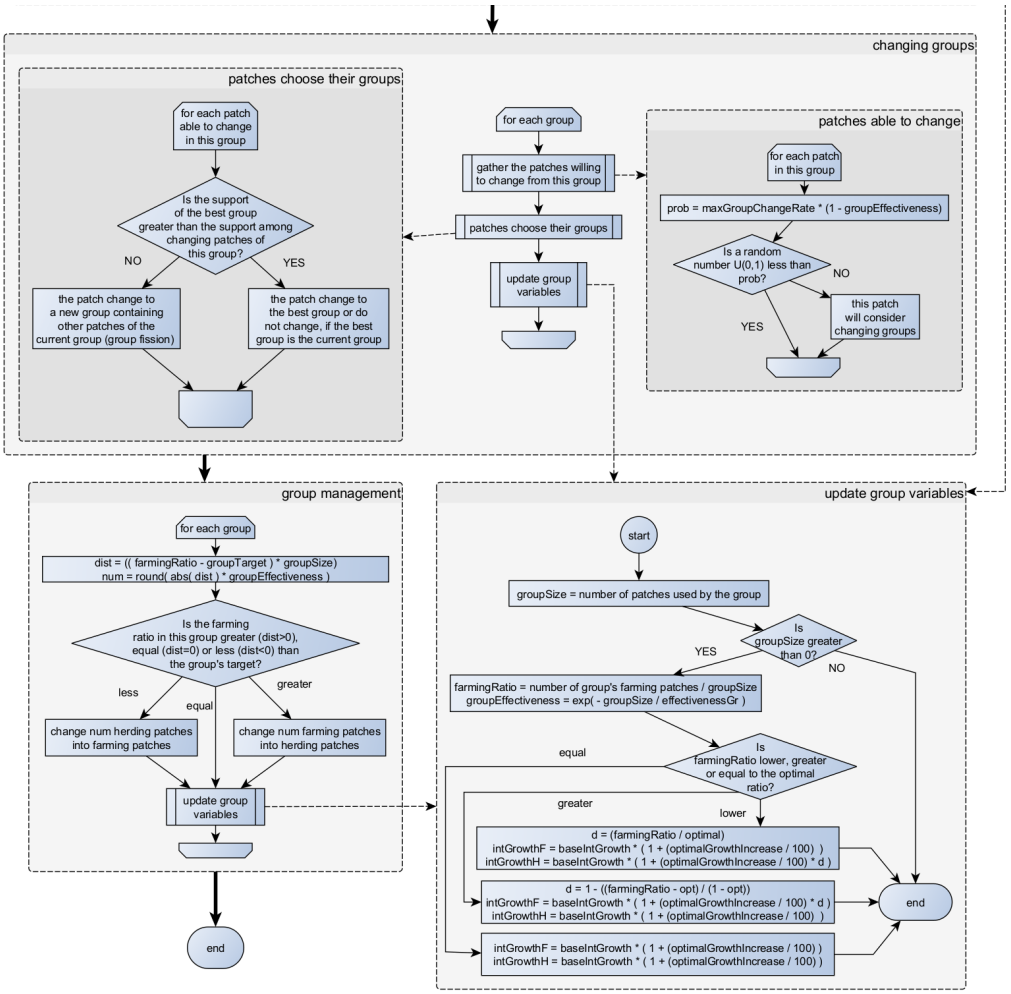
Los apéndices de Angourakis et al. (2017) son los siguientes:

- Appendix A: Diagrama de flujo detallado del ciclo del modelo NMC
- Appendix B: Análisis de sensibilidad sobre el modelo NMC


The Nice Musical Chairs model
start
growth
farming expansion
herding expansion
check competitions
changing groups
group management
end
start
Growth
start
intrinsic growth
extrinsic growth
end
intrinsic growth
for farming and herding
for each group
Is the group intrinsic growth rate for this land use class greater than or equal to a random number (U[0,1])?
YES
NO
+1 groupDemand<F,H>
extrinsic growth
start
farming:
n = round(extGrowth * (total_patches - countLandUseH))
herding:
n = round(extGrowth * (total_patches - countLandUseF))
randomly assign extrinsic growth to groups
end
randomly assign extrinsic growth to groups
repeat n
select a random group
+1 groupDemand<F,H>


land use expansion
farming expansion
start
for each group with groupDemandF > 0
groupDemandRemain = groupDemandF
for each groupDemandRemain (sum of all groups)
Select a random group with groupDemandRemain > 0
Any free land?
NO
YES
Select a random patch
Select a free land unit to be a farming patch associated with this group
Is it already used by this group?
YES
NO
Is it a farming patch?
This group becomes a contender in a F-F competition
YES
NO
Is there a restricted access to pastures?
This group becomes a contender in a F-H competition
YES
NO
The pasture patch is occupied by this group's farming activities
- 1 groupDemandRemain
end
start
farming expansion
herding expansion
end
herding expansion
start
open access
Which pasture tenure regime?
restricted
for each group
for each group
herdingInGroup = number of patches used for herding by this group
groupDemandRemain = herdingInGroup + groupDemandH
groupDemandRemain = groupDemandH
all herding patches are now considered free land
for each groupDemandRemain (sum of all groups)
Select a random group with groupDemandRemain > 0
Any free land?
NO
YES
Select a random patch
Select a free land unit to be a herding patch associated with this group
Is it already used by this group?
YES
NO
Is it a herding patch?
This group becomes a contender in a H-H competition
YES
NO
This group becomes a contender in a F-H competition
-1 groupDemandRemain
end


check competitions
start
Any patches with F-F competitions?
NO
Any patches with F-H competitions?
NO
Any patches with H-H competitions?
NO
Any patches with H-F competitions?
NO
for each group
update group variables
end
YES
for each patch with F-F competitions
YES
for each patch with F-H competitions
YES
for each patch with H-H competitions
YES
for each patch with H-F competitions
for each farming contender in this patch
for each herding contender in this patch
defender: group currently using the patch
resolve competition
resolve competition
start
support = groupSize * groupEffectiveness
ratio = support of contender / (support of contender + support of defender)
Is a random number U(0,1) less than the ratio?
NO
YES
contender is now using this patch
F-F: the land use does not change
H-H: the land use does not change
F-H: the land use becomes farming
H-F: the land use becomes herding
end


changing groups
patches choose their groups
for each patch able to change in this group
Is the support of the best group greater than the support among changing patches of this group?
NO
YES
the patch change to a new group containing other patches of the current group (group fission)
the patch change to the best group or do not change, if the best group is the current group
for each group
gather the patches willing to change from this group
patches choose their groups
update group variables
patches able to change
for each patch in this group
prob = maxGroupChangeRate * (1 - groupEffectiveness)
Is a random number U(0,1) less than prob?
NO
YES
this patch will consider changing groups
group management
for each group
dist = ((farmingRatio - groupTarget) * groupSize)
num = round(abs(dist) * groupEffectiveness)
Is the farming ratio in this group greater (dist>0), equal (dist=0) or less (dist<0) than the group's target?
less
equal
greater
change num herding patches into farming patches
change num farming patches into herding patches
update group variables
end
update group variables
start
groupSize = number of patches used by the group
Is groupSize greater than 0?
YES
NO
farmingRatio = number of group's farming patches / groupSize
groupEffectiveness = exp(- groupSize / effectivenessGr)
Is farmingRatio lower, greater or equal to the optimal ratio?
equal
greater
lower
d = (farmingRatio / optimal)
intGrowthF = baseIntGrowth * (1 + (optimalGrowthIncrease / 100))
intGrowthH = baseIntGrowth * (1 + (optimalGrowthIncrease / 100) * d)
d = 1 - ((farmingRatio - opt) / (1 - opt))
intGrowthF = baseIntGrowth * (1 + (optimalGrowthIncrease / 100) * d)
intGrowthH = baseIntGrowth * (1 + (optimalGrowthIncrease / 100))
intGrowthF = baseIntGrowth * (1 + (optimalGrowthIncrease / 100))
intGrowthH = baseIntGrowth * (1 + (optimalGrowthIncrease / 100))
end

Appendix B:
Sensitivity Analysis on the Nice Musical Chairs model

Overview

This document offers detail on the behavior of the Musical Chairs model under the different conditions explored. In addition to the detailed results of the main design of the model, we present results obtained by performing the same experiments on a second version of the model, where there is no within-class competition. The comparison of these two versions allowed us to dismiss the presence of within-class competition as a relevant factor in producing the differences between the conditions explored.

Logic of the exploration

We followed the steps of Santos et al. (2015) to perform a robust exploration of the model's parameter space and evaluate the influence of each parameter on the state variables.

First, we applied the Latin Hypercube Sampling (LHS) technique to set up our simulation experiments (McKay et al., 1979). This procedure produces quasi-random values for each parameter, sampling the parameter space more homogeneously than other techniques available (B.1).

We assessed the importance of each of the parameters of the model on each of the two main state variables, i.e. the percentage of farming and the size of the biggest group. The parameter importance is here expressed by the Mean Standard Error (MSE) increase when attempting to predict the end value of the state variable without each permuted parameter (Breiman, 2001). To do that, we performed random forest analyses for each of the eight scenarios explored, using the "randomForest" package (Liaw and Wiener, 2002) implemented in R (R Core Team, 2015) and sampling 2000 trees per analysis (ntree=2000). Combining the results obtained with Random Forest analysis with bivariate scatter plots we interpreted the influence of each parameter in terms of absolute differences in the state variables.

Moreover, to render more clearly the influence of restricted access, management and pairing in the model, we performed an additional Random Forest analysis where these three aspects are included as binomial parameters in a data set encompassing the end values under all scenarios.

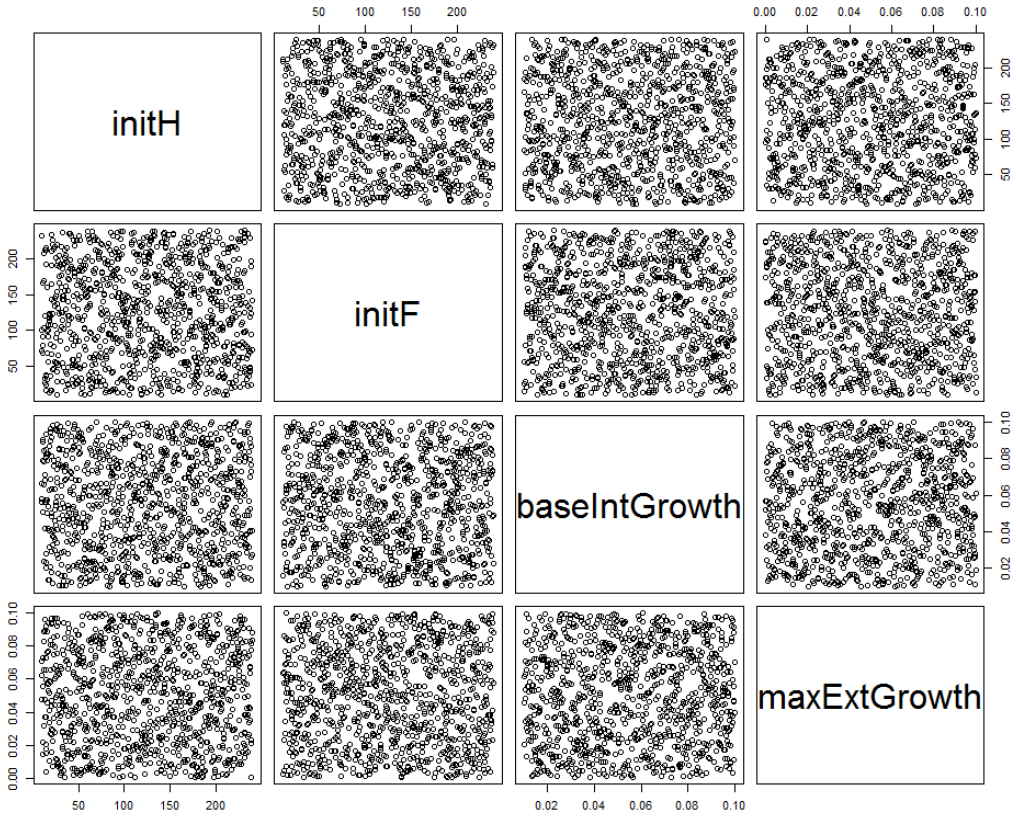


Figure B.1 An example of the sampling of the model's parameter space with LHS (N = 1000). Only four parameters are displayed, for the sake of visualization.

The effect of parameters

The initial number of farming and herding land units

The parameters determining the initial share of land dedicated to farming and herding (init_farming and init_herding) may generally be dismissed as influent factors. The scenario-wise Random Forest analyses in relation to the proportion of farming indicate an exception in scenario Ar (B.2). However, even within this scenario, the

effect of these parameters can be described as a slight offset, in Ar towards more farming (greater init_farming) or more herding (init_herding) against a much stronger tendency towards balanced land use patterns (B.3).

Intrinsic and extrinsic growth rates

Overall, the growth rates have a significant impact on the configuration of land use patterns only in the absence of management (Ao, Ar, Bo and Br, B.2). Both parameters are marked as especially important under an open access regime (Ao and Bo), under which the influence of the extrinsic growth rate is the most significant. Regarding their relationship with the size of the biggest group, the extrinsic growth rate is also the one noteworthy, though in this case particularly under the combination of management and open access (Co and Do, B.3).

Whenever significant, a greater intrinsic growth rate improves the share of farming and strengthens centralization, while the increment of the extrinsic growth rate push results towards a balanced and fragmented pattern (B.4 and B.5).

Although prominent among parameters, the growth rates have mild effects in comparison to the differences among scenarios.

Initial diversity of groups

The initial number of groups is a parameter with broadly no significant effect on the final values observed. A discernable exception occurs in relation to the size of the biggest group under scenarios with management and open access (Co and Do), where the initial group diversity have a slight negative influence over centralization (B.3 and B.5).

Group effectiveness gradient

Generally, the effectiveness gradient is an important parameter of the model. Most clearly, it is the main constraint on the size of the biggest group under all scenarios (B.3 and B.5), as intended in the model design. Concerning the configuration of land use, this parameter is also relatively relevant, though to a lesser degree. Its influence is especially noticeable whenever there is no management and access to pasture is open (Ao and Bo, B.2 and B.4). Nonetheless, the small scale of the variation

produced suggests that the final share of farming land use is broadly independent of the constraints of centralization.

The maximum rate of group change

This parameter constrains the potential migration of individual land units between groups. As expected, it affects mainly the size of the biggest group (B.3), slightly facilitating or obstructing centralization depending on the presence of management (B.5). Moreover, when management and open access are combined (Co and Do), this parameter is the most relevant in configuring the land use pattern (B.2), explicably since the most probable state under these scenarios is the fragmentation of land use in multiple groups. Specifically, in this case, reducing group changing facilitates the emergence of a dominant group specialized in farming (B.4).

Optimal farming ratio and growth increase at the optimum

As anticipated, when pairing is enabled (Bo, Br, Do and Dr), the optimal farming ratio is among the most important parameters in defining the percentage of farming (B.2). Its role is most significant when there is no management and access to pastures is restricted (Br). However, in absolute terms, the effect caused is relatively small, even under scenario Br, where at least some end values can be found for any optimal farming ratio (B.4).

In contrast, the growth increase at the optimum always has a weaker effect on the farming ratio, in both relative and absolute terms.

Finally, the size of the biggest group is broadly non-affected by different values of optimal farming ratio and growth increase at the optimum (B.3 and B.5).

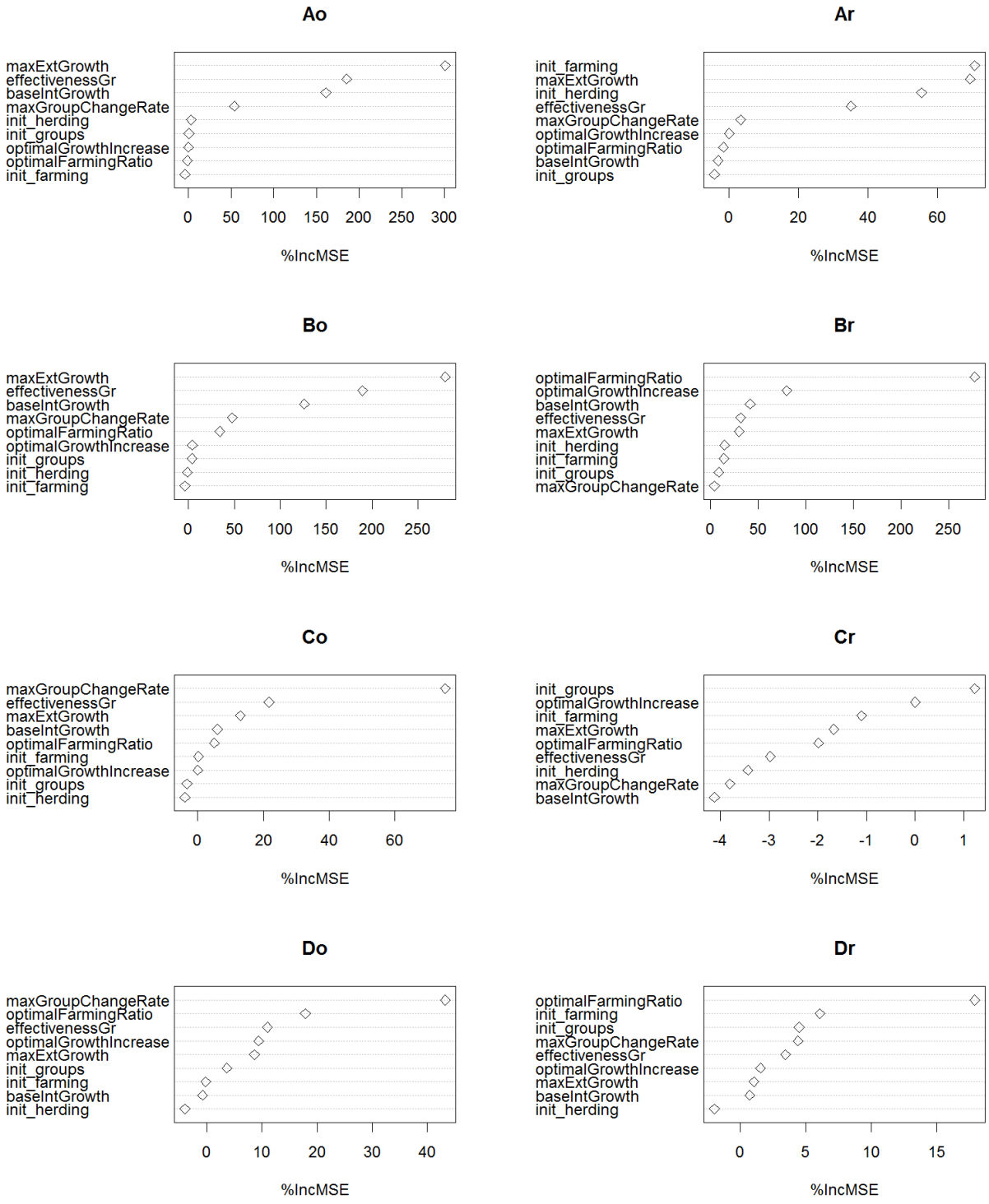


Figure B.2: The ranked parameter's importance in respect to farming across all experiments

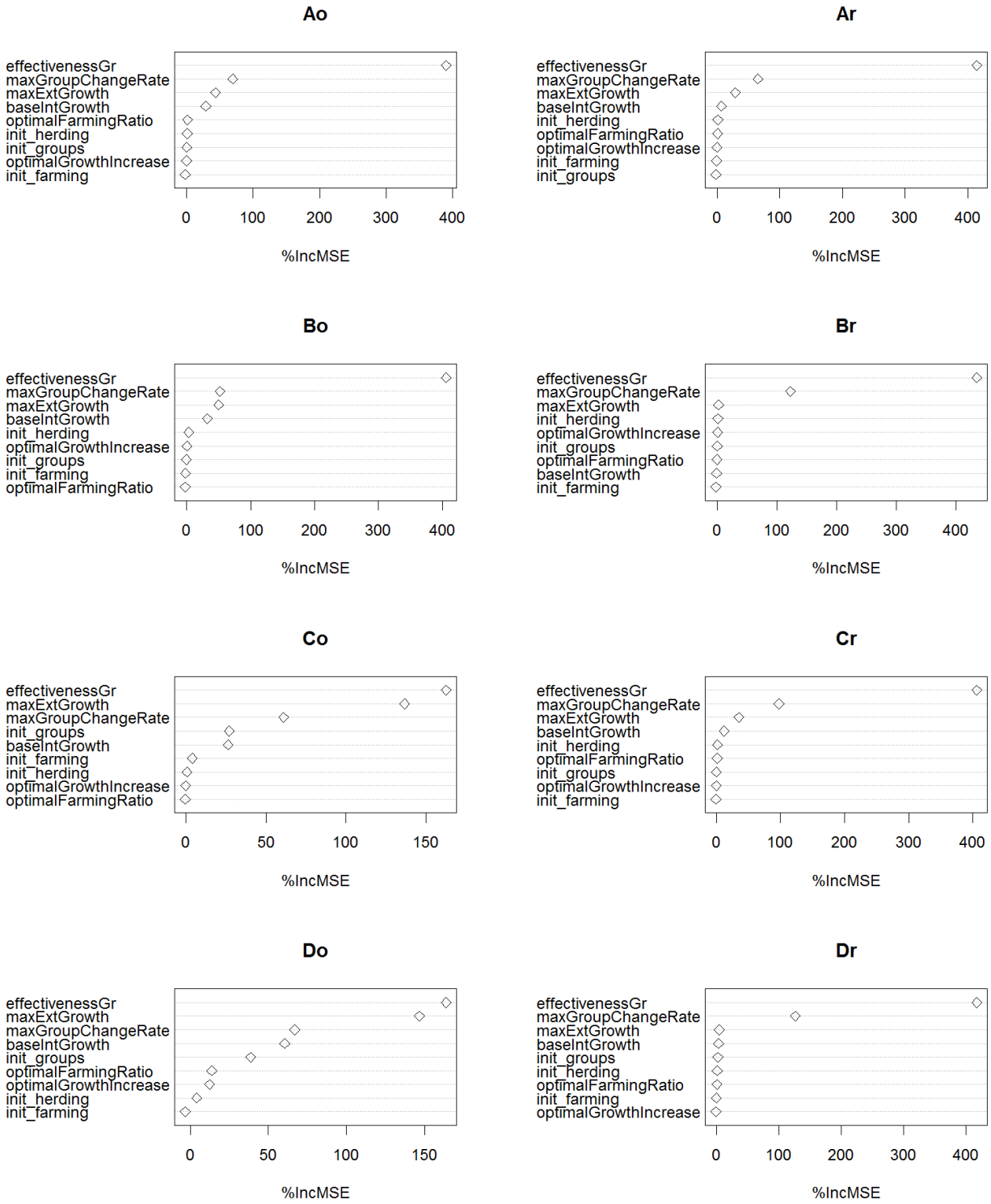


Figure B.3: The ranked parameter's importance in respect to the size of the biggest group across all experiments

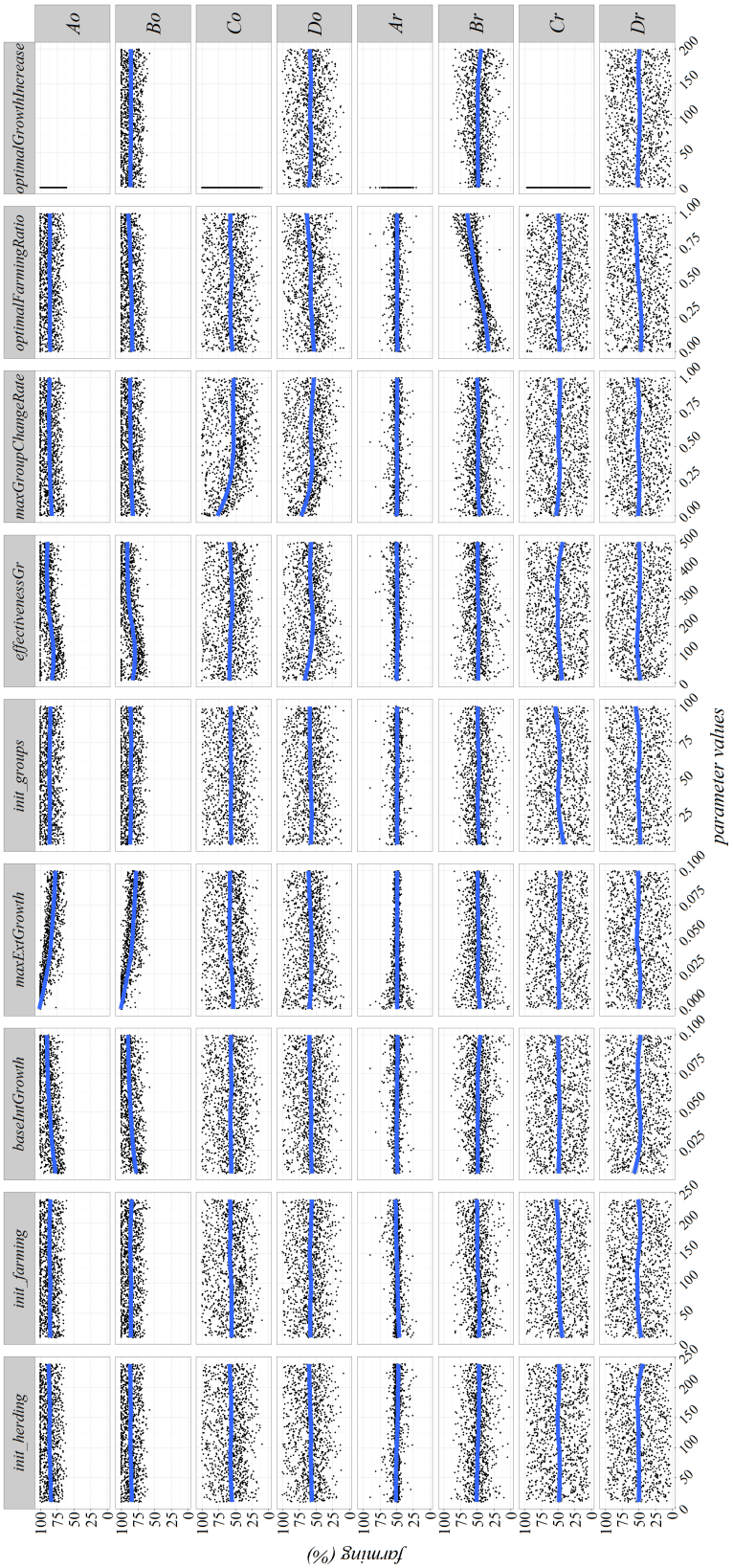


Figure B.4: The extent of farming according to scenario and parameters' values

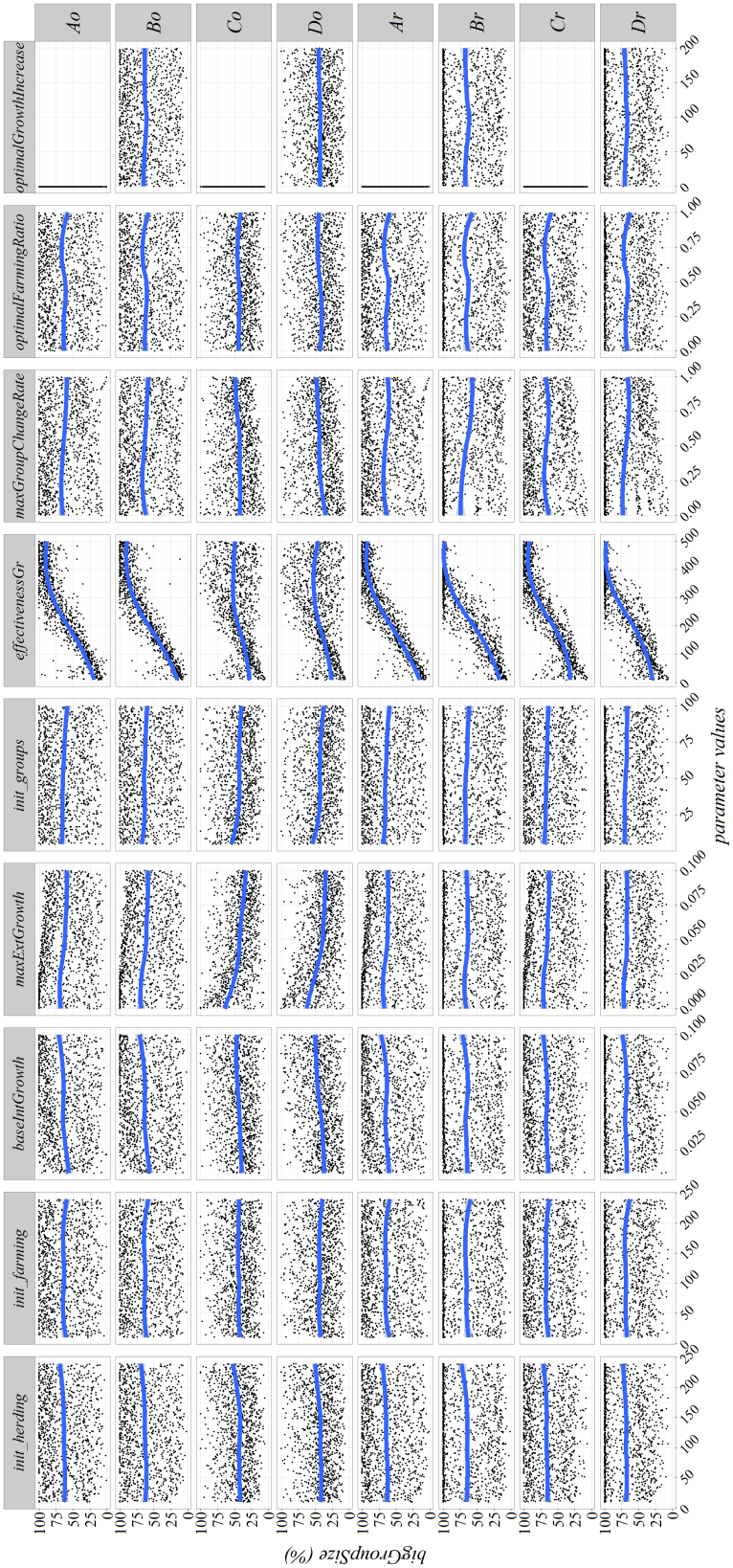


Figure B.5: The the size of the biggest group according to scenario and parameters' values

Version without within-class competition

Given the introduction of within-class competition (farming-farming and herding-herding competition), which was absent in the Musical Chairs model, we deemed relevant to evaluate its effect, especially in contrast with the variations observed between scenarios. For this purpose, all sets of experiments were repeated consenting only between-class competition (farming disputing herding land units, and vice-versa).

The effect of scenarios

Regarding the differences between scenarios, only minor variations are appreciable in respect to the main version of the model (B.6 and B.7). Essentially, the presence of within-class competition eases centralization, explicably because groups compete more intensively, disregarding their internal farming ratio. In this sense, when this aspect is excluded from the model, the system dynamic will more frequently be trapped in local equilibria where specialized groups coexist. The amplitude of this phenomenon is greater with open access to pastures and no management (Ao and Bo), when trajectories converge either in fragmented more diversified land use patterns or the complete centralization and specialization in farming. In contrast, under scenarios Cr and Dr, within-class competition is of little consequence to the long-term land use pattern.

This version is consistent with the main version of the model also when considering the relative importance of the three aspects varying between scenarios (B.8). The most important aspect in defining the percentage of farming is the regime of access to pasture, followed by management, while the size of the biggest group is most affected by the effectiveness gradient, and again by the regime of access to pastures and management.

The effect of parameters

Without within-class competition, most parameters hold the same importance and produce the same effects described above regarding the main version (B.9, B.10, B.11, and B.12).

The highlights and exceptions are the following:

- The **initial number of farming and herding land units** are generally unimportant, except under the scenario Ar, where these parameters produce a slight unbalance in the otherwise balanced farming ratio (B.9 and B.11).
- The **intrinsic and extrinsic growth rates** are generally less important than they are in the main version (B.9 and B.10), explicably because part of the pressure generated by growth is dismissed when within-class competition is not possible.
- In contrast with the main version, the **initial number of groups** has a mild but relevant effect over the percentage of farming with open access to pastures (B.9), especially without management (Ao and Bo), while it has still a significant influence on the size of the biggest group under the scenarios Co and Do (B.10 and B.12).
- As in the main version, the **group effectiveness gradient** is one of the most influent parameters of the model, especially under scenarios Ao and Bo where it heavily impacts the percentage of farming (B.9 and B.11).
- The **maximum rate of group change** has exactly the same effect and importance when there is no within-class competition (B.9 and B.10).
- The two parameters regulating pairing, the **optimal farming ratio** and the **growth increase at the optimum**, are even less important without within-class competition (B.9 and B.10). The exception occurs under scenario Br regarding the proportion of farming; however, as in the main version, the absolute effect is quite small (B.11).

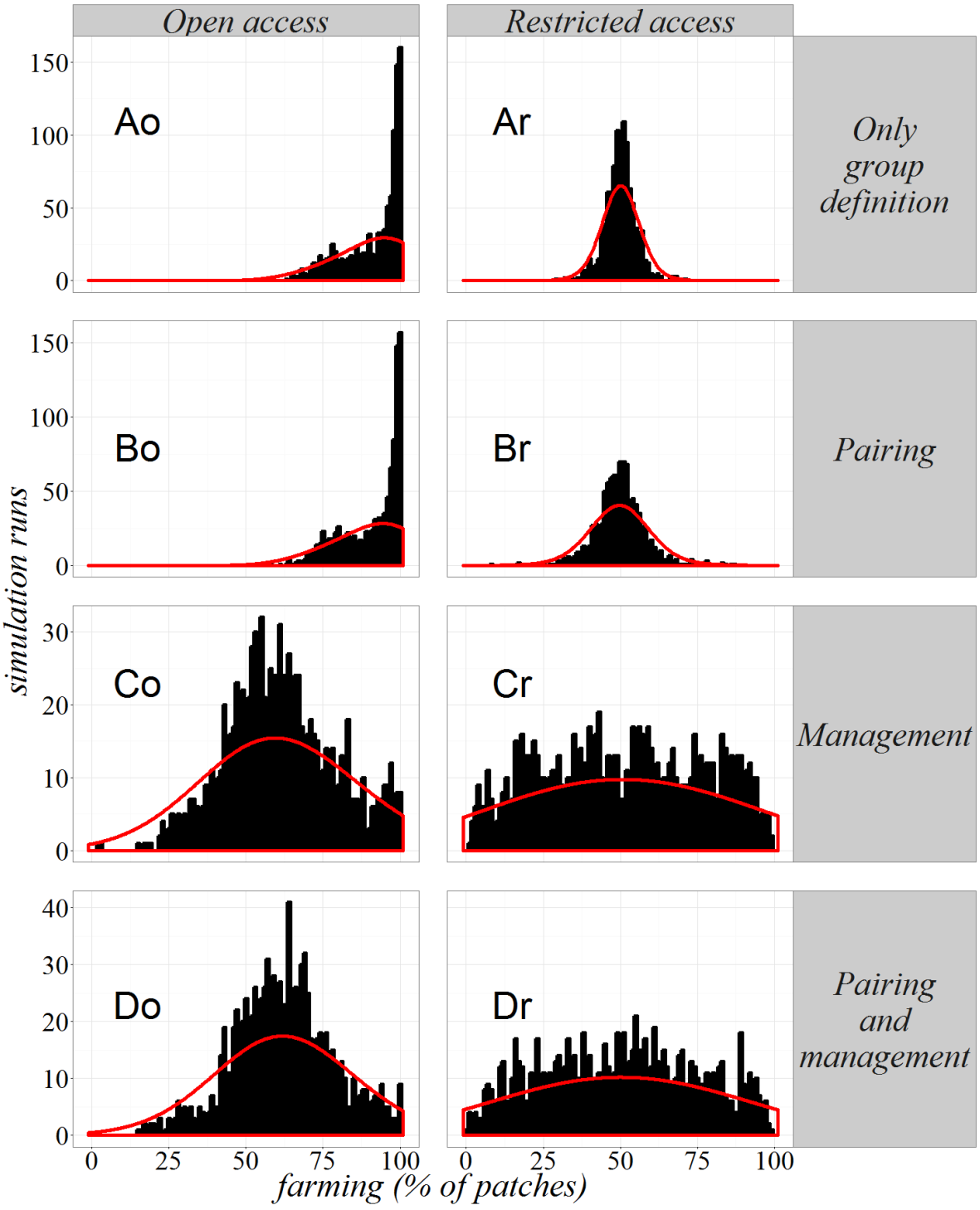


Figure B.6: The count of simulation runs stabilizing at different extents of farming and the respective density projections (lines) for each of the eight scenarios explored in experiments without intra-class competition

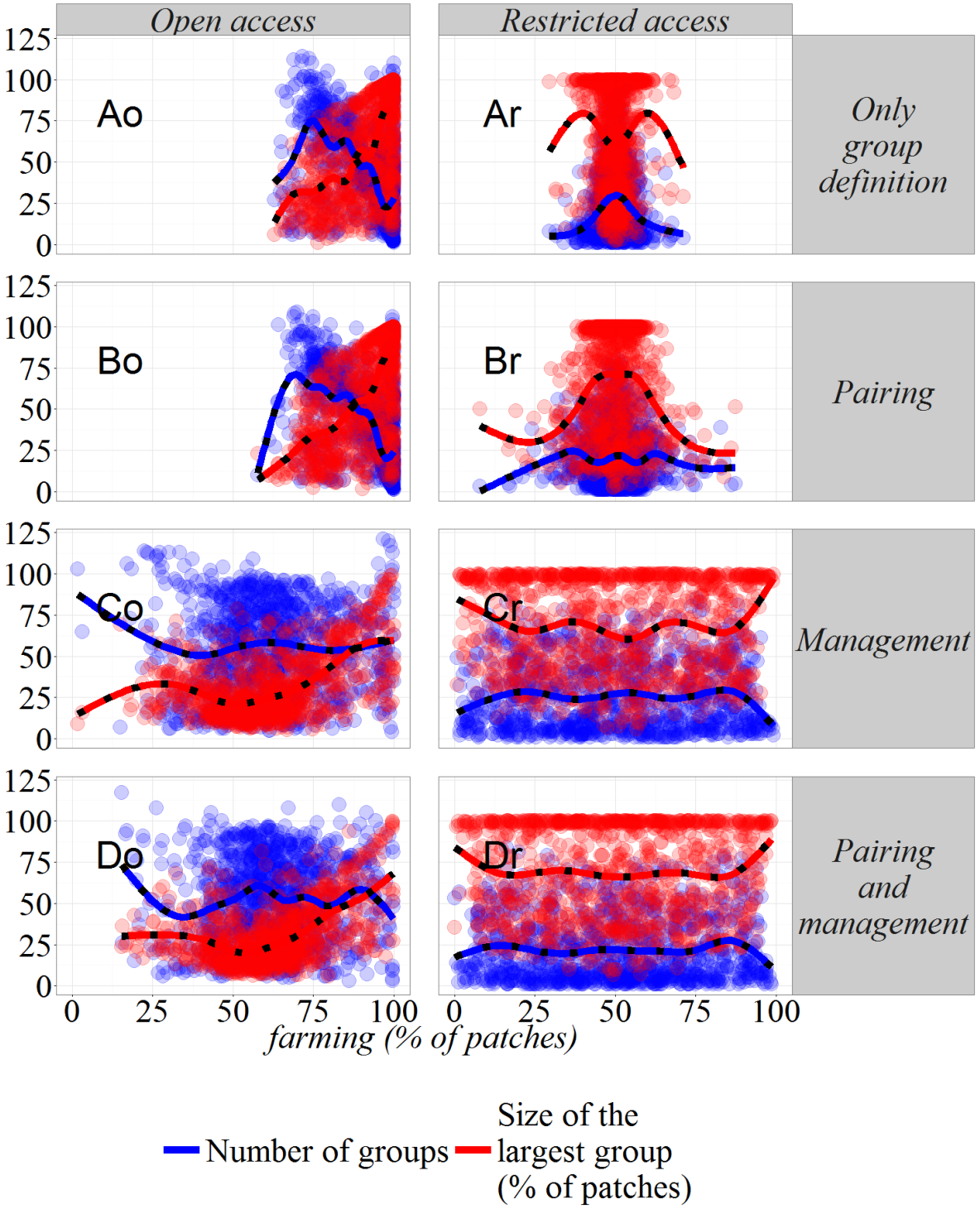


Figure B.7: The extent of farming versus the number of groups and the size of the biggest group. The lines represent the generalized additive model (GAM), using a cubic regression spline, for the each variable in experiments without intra-class competition

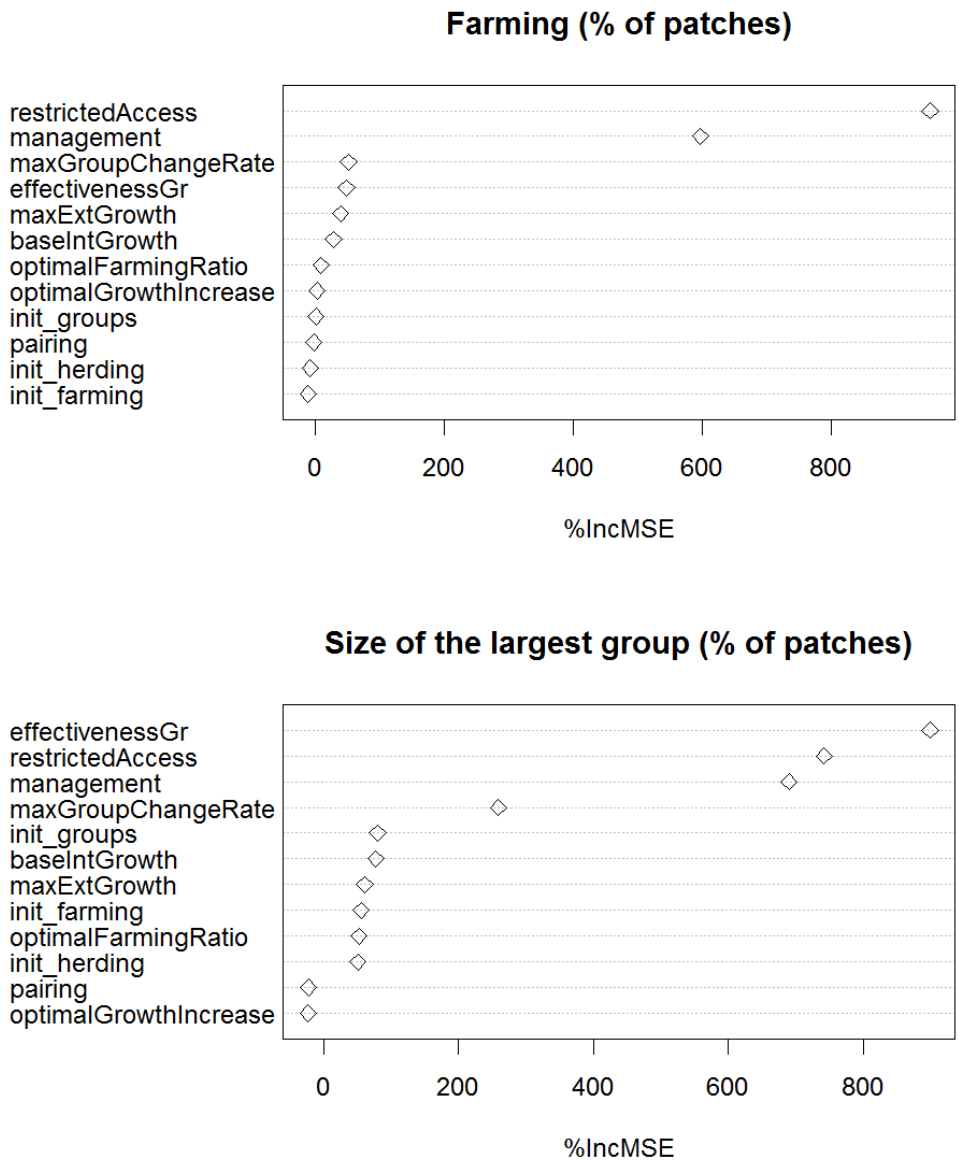


Figure B.8: The ranked parameter's importance in respect to farming and the size of the biggest group across all experiments without intra-class competition

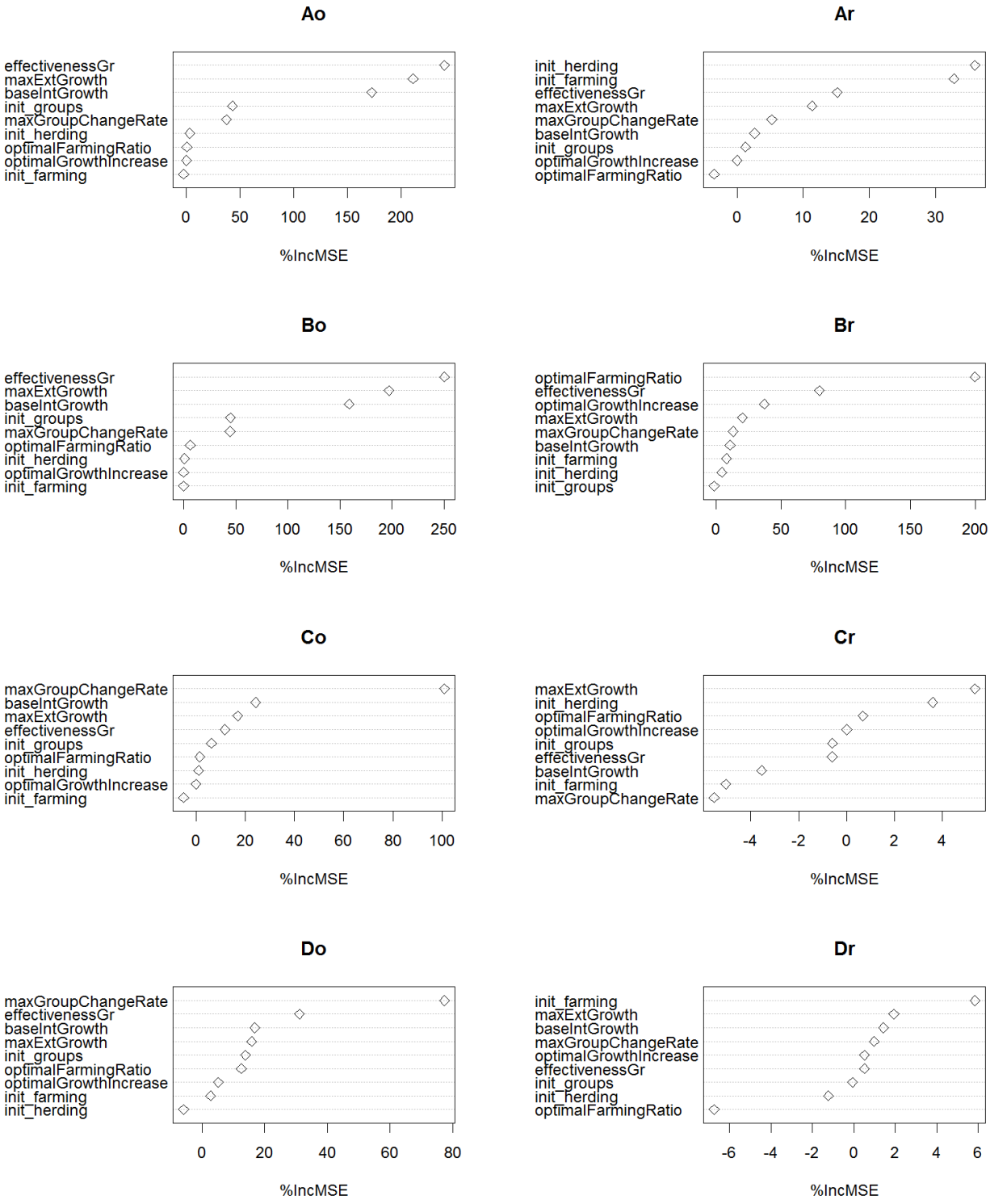


Figure B.9: The ranked parameter's importance in respect to farming across all experiments without intra-class competition.

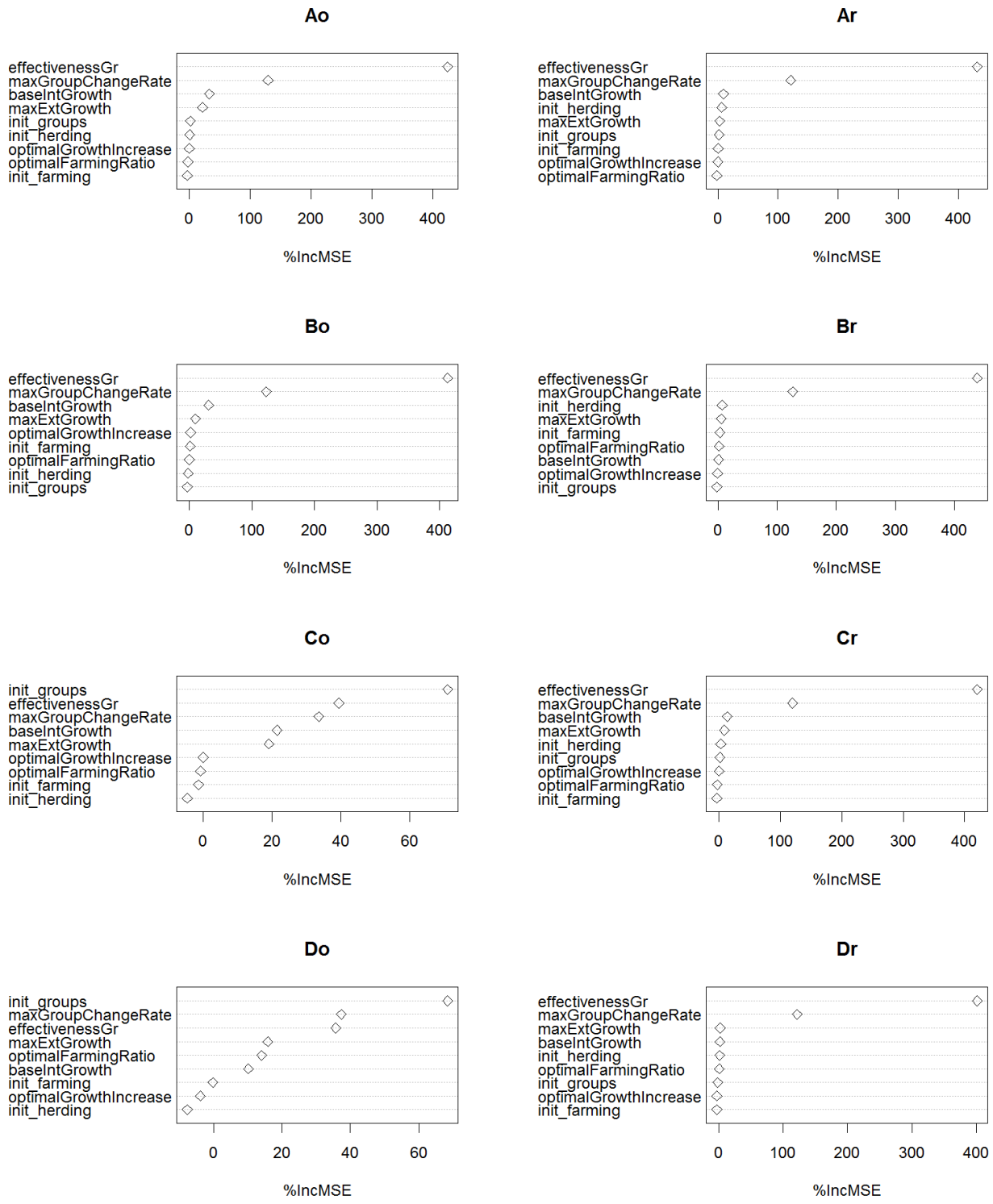


Figure B.10: The ranked parameter's importance in respect to the size of the biggest group across all experiments without intra-class competition

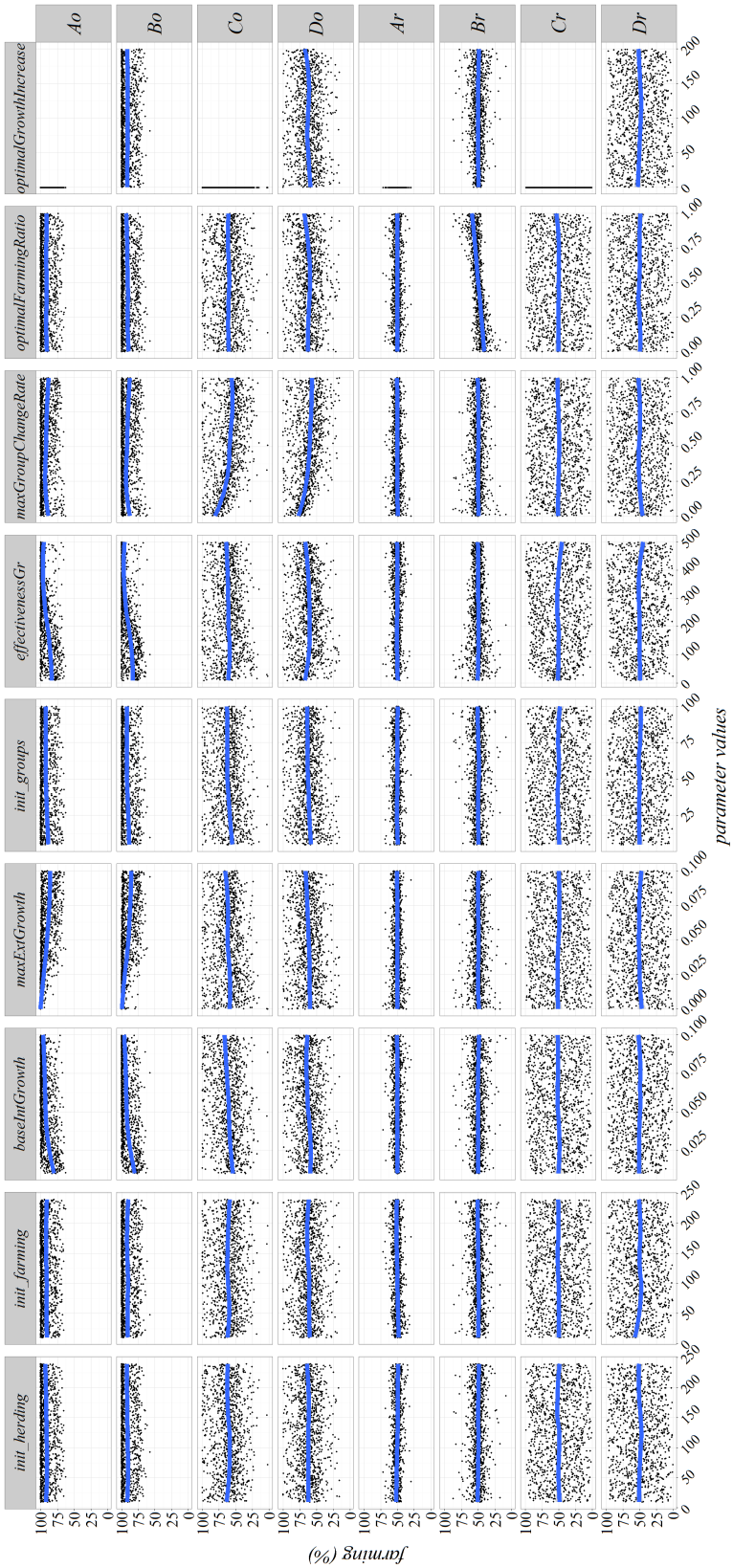


Figure B.11: The extent of farming according to scenario and parameters' values

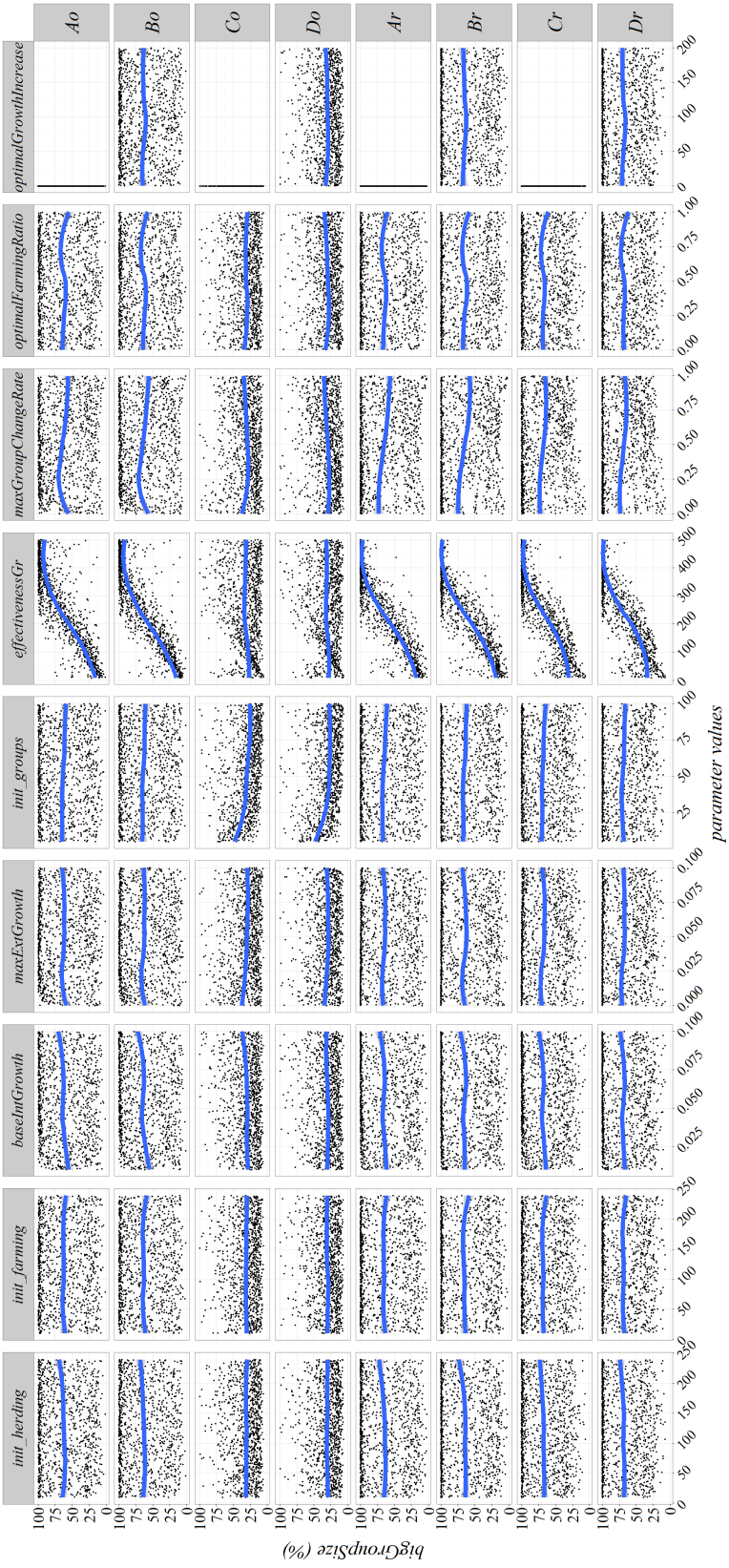


Figure B.12: The the size of the biggest group according to scenario and parameters' values

References

Breiman, L., 2001. Random Forests. Mach. Learn. 45, 5–32. doi:10.1023/A:1010933404324

Liaw, A., Wiener, M., 2002. Classification and Regression by randomForest. R News 2, 18–22.

McKay, M.D., Beckman, R.J., Conover, W.J., 1979. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. Technometrics 21, 239–245. doi:10.2307/1268522

R Core Team, 2015. R: A language and environment for statistical computing.

Santos, J.I., Pereda, M., Zurro, D., Álvarez, M., Caro, J., Galán, J.M., Briz i Godino, I., 2015. Effect of resource spatial correlation and hunter-fisher-gatherer mobility on social cooperation in Tierra del Fuego. PLoS One 10, e0121888. doi:10.1371/journal.pone.0121888

5.3.7. A.2.3 Apéndices de 'Food For All: An Agent-Based model to explore the emergence and implications of cooperation for food storage' { - }

Los apéndices de Angourakis et al. (2015) son los siguientes:

- Supplementary Material 1: Los parámetros y variables del modelo
- Supplementary Material 2: Las conyecturas referentes al almacenamiento de comida en "Food For All"
- Supplementary Material 3: La cantidad de comida almacenable adquirida por períodos
- Supplementary Material 4: El estímulo de aprendizaje según el número de períodos sin escasez
- Supplementary Material 5: Las medias de cooperación para una norma de cooperación intermedia ($\rho = 0{,}5$) y niveles bajos de estímulo de aprendizaje ($T_h = 10, 20$)
- Supplementary Material 6: Las desviaciones típicas de cooperación para una norma de cooperación intermedia ($\rho = 0{,}5$) y niveles bajos de estímulo de aprendizaje ($T_h = 10, 20$)
- Supplementary Material 7: Las desviaciones típicas de cooperación para niveles moderado ($T_h = 30$) y alto ($T_h = 40$) de estímulo de aprendizaje

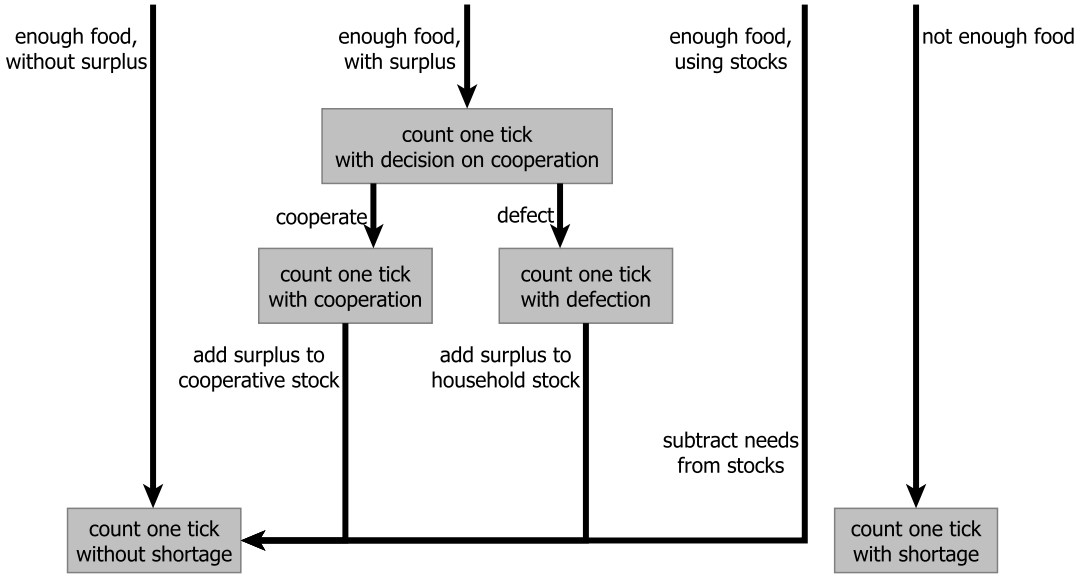
Supplementary Material 1: parameters and variables of the model

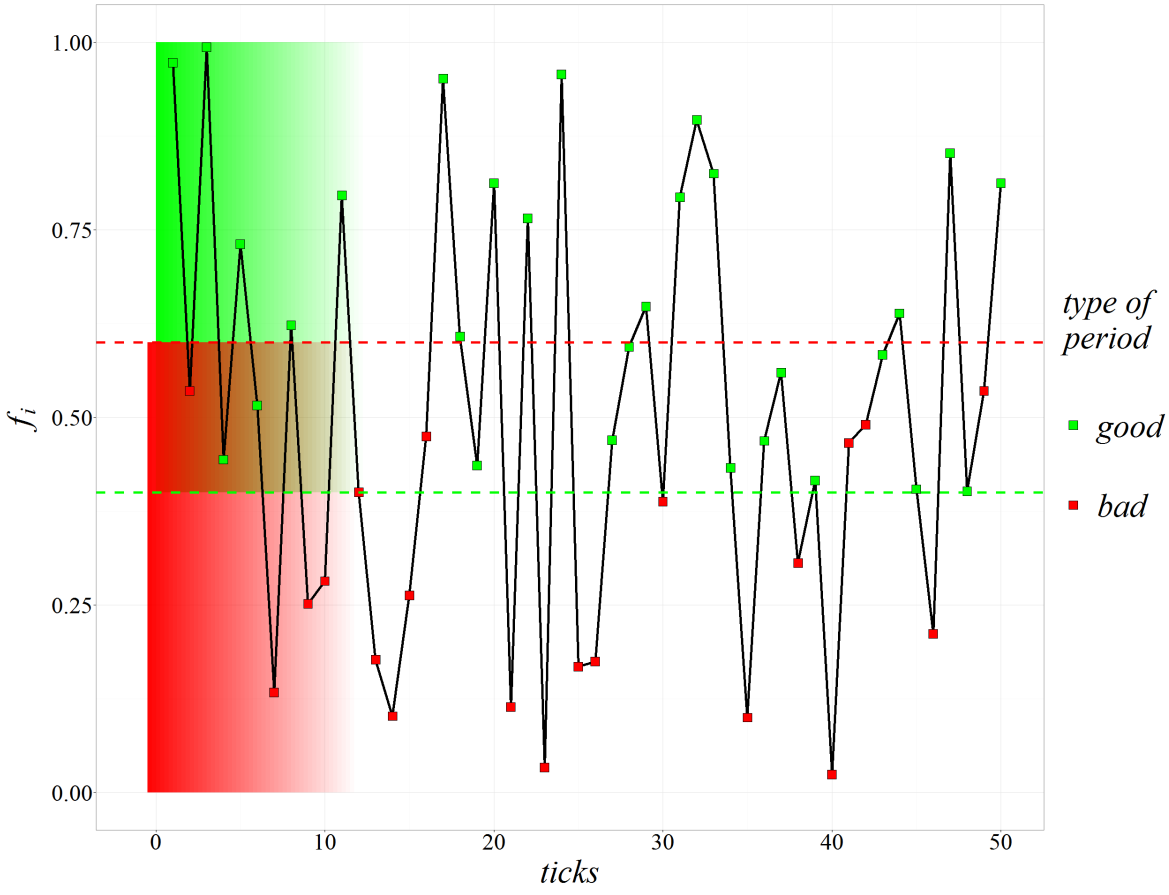
Table S1.1: Agents' state variables	
Parameter name	**Brief description**
cooperation-probability (p_i)	Agent's probability to cooperate
private-storage ($S_{v,i}$)	Accumulated food in household's stock
n-non-shortage-ticks	Number of time periods in a generation in which an agent do not suffers from shortage
n-decision-ticks	Number of time periods in a generation in which an agent makes a decision regarding storage (to cooperate or to defect)
n-cooperation-ticks	Number of time periods in a generation in which an agent decides to cooperate

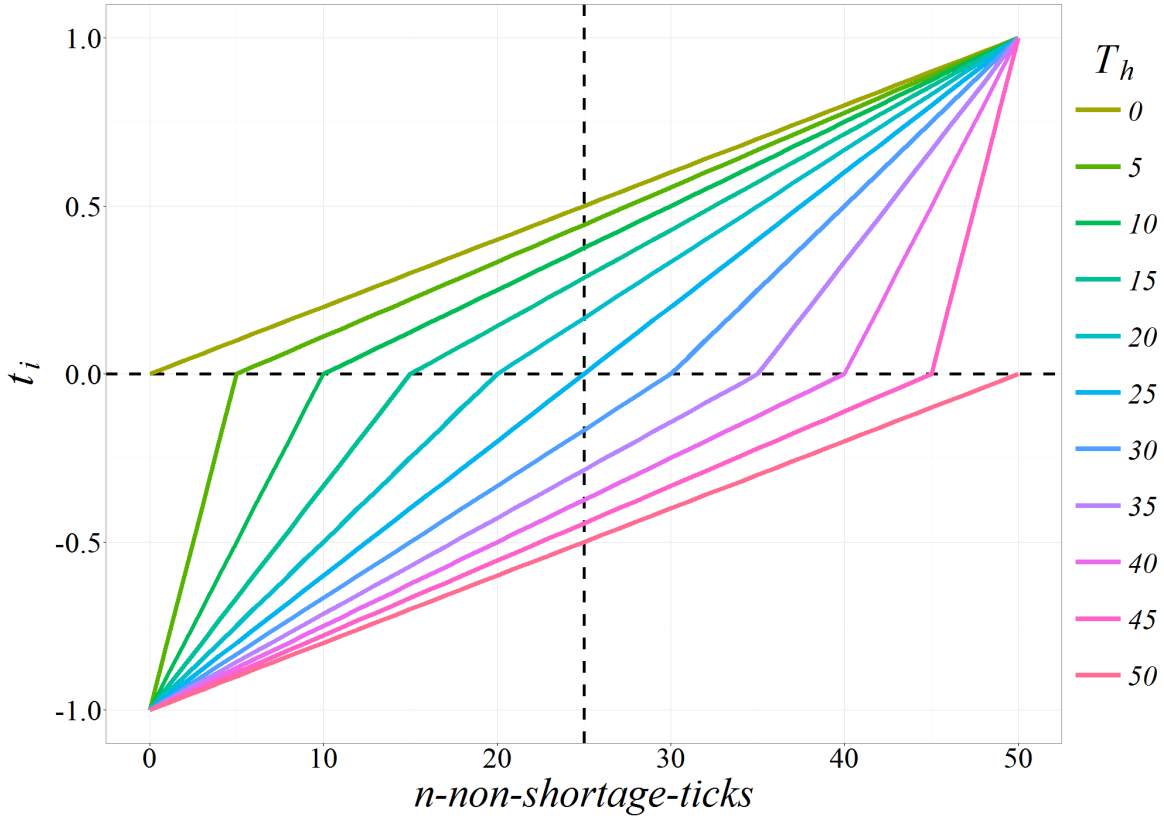
Table S1.2: Constants		
Parameter name	**Brief description**	**Value**
n-people (N)	The number of households in the population	100
U^{good}	Probability distribution of household productivity in a good period	U(0.4-1)
U^{bad}	Probability distribution of household productivity in a bad period	U(0-0.6)
p_{good} / p_{bad}	The probability that a period is good/bad	0.5
learning-generation	The number of time periods between the re-assessment of agents' strategies	50
shortage-threshold ($\bar{S}$)	The amount of food each household needs during one time period. An agent not satisfying this need is considered to suffer from shortage	0.5
learning-rate (L)	The pace in which an agent modifies its strategy	0.7

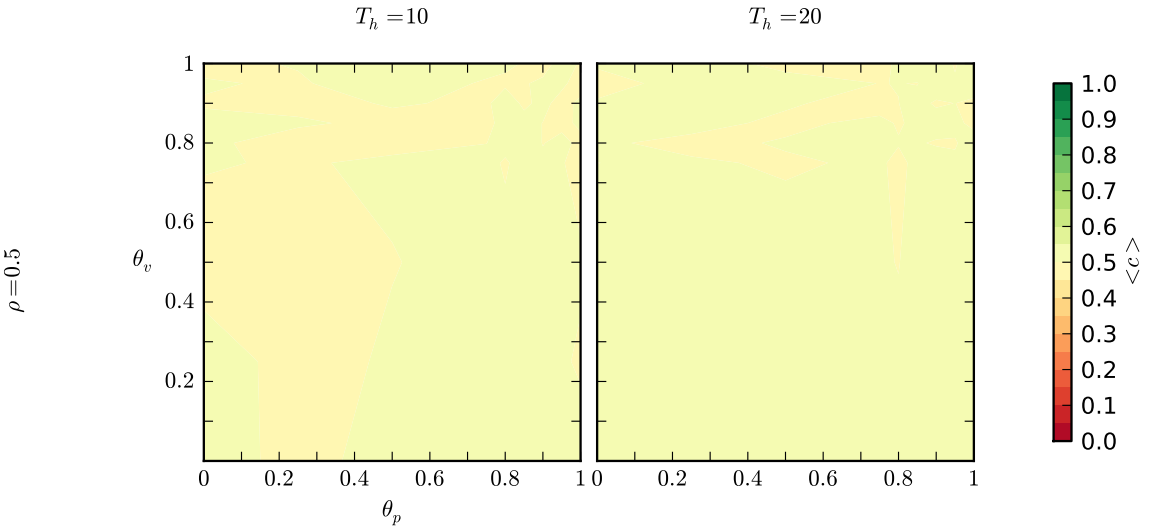
Table S1.3: Study parameters	
Parameter name	**Brief description**
public-storage-efficiency (θ_p)	The fraction of the cooperative stock preserved for the next time period
private-storage-efficiency (θ_v)	The fraction of households' stocks preserved for the next time period
aspiration-threshold (T_h)	The number of time periods in a generation beyond which a people agent considers the frequency of shortage to be unacceptable (see Learning process)
cooperation-rate-required (ρ)	The rate of cooperation per generation required to permit access to the cooperative storage (i.e., with $\rho = 1$, access is only permited for fully-cooperative agents)

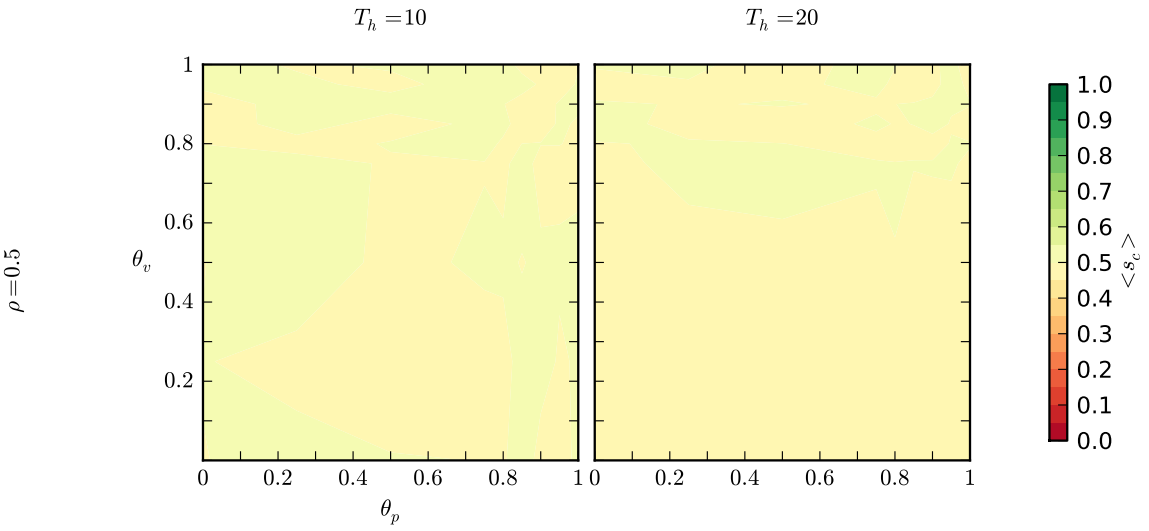
Table S1.4: Global state variables	
Parameter name	**Brief description**
$< c >$	the average cooperation of the strategies of agents
$< S_c >$	the standard deviation of cooperation of the strategies of agents
public-storage (S_p)	The quantity of food accumulated in cooperative stock
season-period	Type of season that determines the probability distribution of food at each time period, i.e. "good" or "bad"

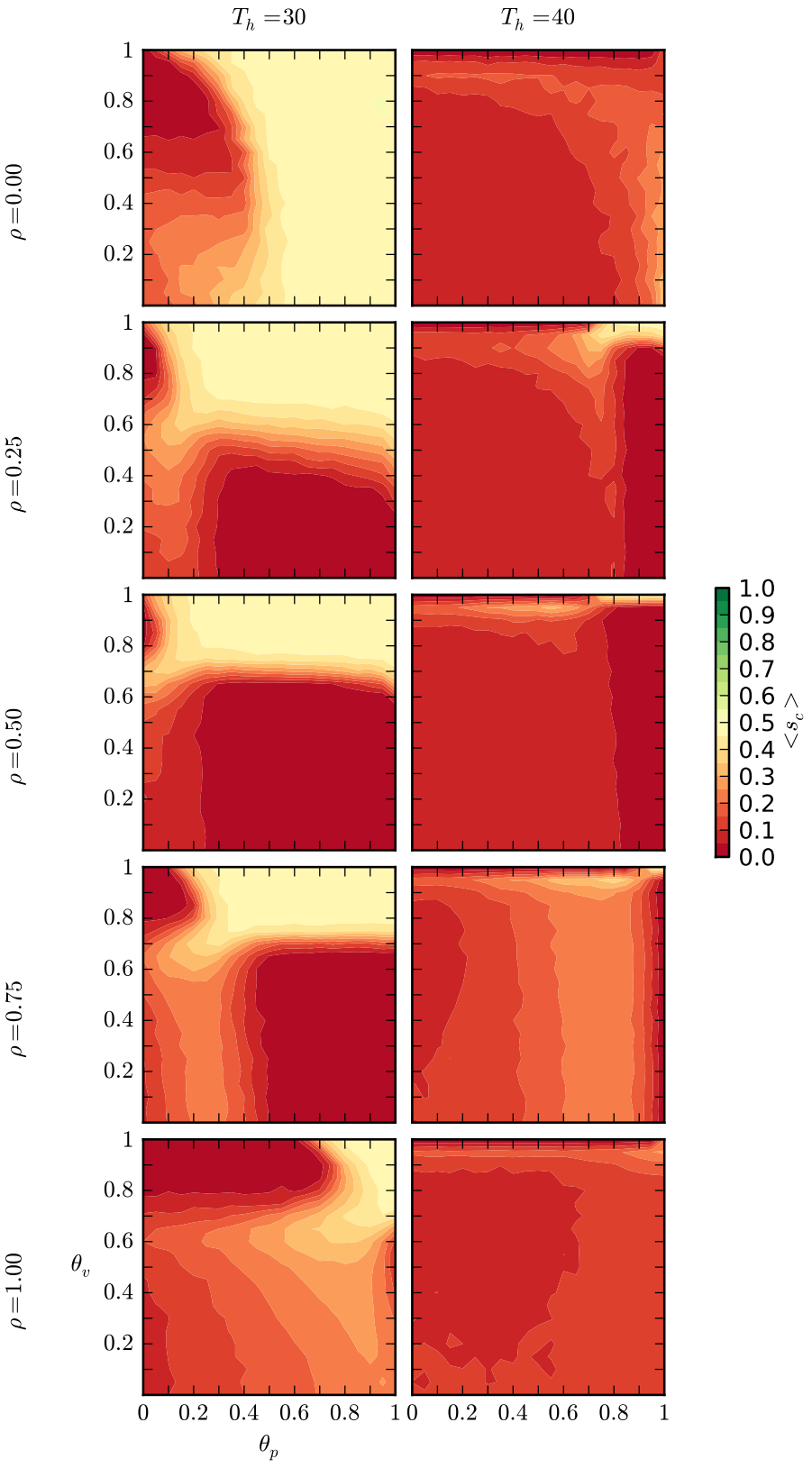

enough food,
without surplus
enough food,
with surplus
enough food,
using stocks
not enough food
count one tick
with decision on cooperation
cooperate
defect
count one tick
with cooperation
count one tick
with defection
add surplus to
cooperative stock
add surplus to
household stock
subtract needs
from stocks
count one tick
without shortage
count one tick
with shortage


1.00
0.75
0.50
0.25
0.00
f_i
0
10
20
30
40
50
ticks
type of
period
good
bad


1.0
0.5
0.0
-0.5
-1.0
t_i
0
10
20
30
40
50
n-non-shortage-ticks
T_h
0
5
10
15
20
25
30
35
40
45
50


T_h =10
T_h =20
ρ=0.5
θ_v
θ_p
1
0.8
0.6
0.4
0.2
0
0.2
0.4
0.6
0.8
1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
<c>


T_h =10
T_h =20
ρ=0.5
θ_v
θ_p
0
0.2
0.4
0.6
0.8
1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
$< s_c >$


T_h =30
T_h =40
ρ=0.00
ρ=0.25
ρ=0.50
ρ=0.75
ρ=1.00
θ_v
θ_p
$< s_c >$