

# Manual for Package: mathematics

## Revision 2:14M

Karl Kästner

August 26, 2020

## Contents

<b>1</b>	<b>calendar</b>	<b>33</b>
1.1	days_per_month . . . . .	33
1.2	isnight . . . . .	33
<b>2</b>	<b>mathematics</b>	<b>33</b>
2.1	cast_byte_to_integer . . . . .	33
<b>3</b>	<b>complex-analysis</b>	<b>33</b>
3.1	complex_exp_product_im_im . . . . .	33
3.2	complex_exp_product_im_re . . . . .	34
3.3	complex_exp_product_re_im . . . . .	34
3.4	complex_exp_product_re_re . . . . .	34
3.5	croots . . . . .	35
3.6	root_complex . . . . .	35
3.7	test_imroots . . . . .	35
<b>4</b>	<b>derivation</b>	<b>35</b>
4.1	derive_acfar1 . . . . .	35
4.2	derive_ar2param . . . . .	35
4.3	derive_arc_length . . . . .	35
4.4	derive_fourier_power . . . . .	36
4.5	derive_fourier_power_exp . . . . .	36
4.6	derive_laplacian_curvilinear . . . . .	36
4.7	derive_laplacian_fourier_piecewise_linear . . . . .	36
4.8	derive_logtripdf . . . . .	36
4.9	derive_smooth1d_parametric . . . . .	36
<b>5</b>	<b>derivation/master</b>	<b>36</b>
5.1	derive_bc_one_sided . . . . .	36
5.2	derive_convergence . . . . .	36

5.3	derive_error_fdm . . . . .	36
5.4	derive_fdm_poly . . . . .	37
5.5	derive_fdm_power . . . . .	37
5.6	derive_fdm_taylor . . . . .	37
5.7	derive_fdm_vargrid . . . . .	37
5.8	derive_fem_2d_mass . . . . .	37
5.9	derive_fem_error_2d . . . . .	37
5.10	derive_fem_error_3d . . . . .	37
5.11	derive_fem_sym_2d . . . . .	37
5.12	derive_grid_constants . . . . .	37
5.13	derive_interpolation . . . . .	37
5.14	derive_laplacian . . . . .	38
5.15	derive_limit . . . . .	38
5.16	derive_nc_1d . . . . .	38
5.17	derive_nc_1d_ . . . . .	38
5.18	derive_nc_2d . . . . .	38
5.19	derive_nonuniform_symmetric . . . . .	38
5.20	derive_richardson . . . . .	38
5.21	derive_sum . . . . .	38
5.22	nn . . . . .	38
5.23	test_derive . . . . .	38
5.24	test_derive_fdm_poly . . . . .	39
5.25	test_filter . . . . .	39
5.26	test_vargrid . . . . .	39
<b>6</b>	<b>derivation</b>	<b>39</b>
6.1	simplify_atan . . . . .	39
<b>7</b>	<b>mathematics</b>	<b>39</b>
7.1	entropy . . . . .	39
7.2	exp10 . . . . .	39
7.3	filter_twosided . . . . .	39
<b>8</b>	<b>finance</b>	<b>39</b>
8.1	derive_skewrnd_walsh_paramter . . . . .	39
8.2	gbm_cdf . . . . .	40
8.3	gbm_fit . . . . .	40
8.4	gbm_fit_old . . . . .	40
8.5	gbm_inv . . . . .	40
8.6	gbm_mean . . . . .	40
8.7	gbm_median . . . . .	40
8.8	gbm_pdf . . . . .	40
8.9	gbm_simulate . . . . .	40
8.10	gbm_skewness . . . . .	40

8.11	gbm_std . . . . .	40
8.12	gbm_transform_time_step . . . . .	41
8.13	put_price_black_scholes . . . . .	41
8.14	skewgbm_simulate . . . . .	41
8.15	skewrnd_walsh . . . . .	41
<b>9</b>	<b>finance/test</b>	<b>41</b>
9.1	test_gbm . . . . .	41
9.2	test_gbm_pdf . . . . .	41
9.3	test_skewrnd_walsh . . . . .	41
<b>10</b>	<b>fourier/@STFT</b>	<b>41</b>
10.1	STFT . . . . .	41
10.2	itransform . . . . .	42
10.3	stft_ . . . . .	42
10.4	stftmat . . . . .	42
10.5	transform . . . . .	42
<b>11</b>	<b>fourier</b>	<b>42</b>
11.1	amplitude_from_peak . . . . .	42
11.2	dftmtx_man . . . . .	43
11.3	example_fourier_window . . . . .	43
11.4	fft_derivative . . . . .	43
11.5	fft_man . . . . .	43
11.6	fftsmooth . . . . .	44
11.7	fix_fourier . . . . .	44
11.8	fourier_axis . . . . .	44
11.9	fourier_cesaro_correction . . . . .	45
11.10	fourier_coefficient_piecewise_linear . . . . .	45
11.11	fourier_coefficient_piecewise_linear_1 . . . . .	45
11.12	fourier_coefficient_ramp3 . . . . .	45
11.13	fourier_coefficient_ramp_pulse . . . . .	46
11.14	fourier_coefficient_ramp_step . . . . .	46
11.15	fourier_coefficient_square_pulse . . . . .	46
11.16	fourier_cubic_interaction_coefficients . . . . .	46
11.17	fourier_derivative . . . . .	46
11.18	fourier_expand . . . . .	46
11.19	fourier_fit . . . . .	46
11.20	fourier_interpolate . . . . .	46
11.21	fourier_matrix . . . . .	47
11.22	fourier_matrix2 . . . . .	47
11.23	fourier_matrix3 . . . . .	47
11.24	fourier_matrix_exp . . . . .	47
11.25	fourier_multiplicative_interaction_coefficients . . . . .	47

11.26	fourier_power . . . . .	47
11.27	fourier_power_exp . . . . .	48
11.28	fourier_predict . . . . .	48
11.29	fourier_quadratic_interaction_coefficients . . . . .	48
11.30	fourier_range . . . . .	48
11.31	fourier_regress . . . . .	48
11.32	fourier_resampled_fit . . . . .	48
11.33	fourier_resampled_predict . . . . .	49
11.34	fourier_signed_square . . . . .	49
11.35	fourier_transform . . . . .	49
11.36	hyperbolic_fourier_box . . . . .	49
11.37	idftmtx_man . . . . .	49
11.38	laplace_2d_pwlinear . . . . .	50
11.39	nanfft . . . . .	50
11.40	peaks . . . . .	50
11.41	roots_fourier . . . . .	51
11.42	spectral_density . . . . .	51
11.43	test_complex_exp_product . . . . .	51
11.44	test_fourier_filter . . . . .	51
11.45	test_idftmtx . . . . .	51
<b>12 mathematics</b>		<b>51</b>
12.1	gaussfit_quantile . . . . .	51
<b>13 geometry/@Geometry</b>		<b>51</b>
13.1	Geometry . . . . .	51
13.2	arclength . . . . .	52
13.3	arclength_old . . . . .	52
13.4	arclength_old2 . . . . .	52
13.5	base_point . . . . .	52
13.6	base_point_limited . . . . .	52
13.7	centroid . . . . .	52
13.8	cosa_min_max . . . . .	52
13.9	cross2 . . . . .	53
13.10	curvature . . . . .	53
13.11	ddot . . . . .	53
13.12	distance . . . . .	53
13.13	distance2 . . . . .	53
13.14	dot . . . . .	53
13.15	edge_length . . . . .	53
13.16	enclosed_angle . . . . .	53
13.17	enclosing_triangle . . . . .	54
13.18	hexagon . . . . .	54
13.19	inPolygon . . . . .	54

13.20	inTetra . . . . .	54
13.21	inTetra2 . . . . .	54
13.22	inTriangle . . . . .	54
13.23	intersect . . . . .	54
13.24	lineintersect . . . . .	54
13.25	lineintersect1 . . . . .	55
13.26	minimum_distance_lines . . . . .	55
13.27	mittenpunkt . . . . .	55
13.28	nagelpoint . . . . .	55
13.29	onLine . . . . .	55
13.30	orthocentre . . . . .	55
13.31	plumb_line . . . . .	55
13.32	poly_area . . . . .	55
13.33	poly_edges . . . . .	56
13.34	poly_set . . . . .	56
13.35	poly_width . . . . .	56
13.36	polyxpoly . . . . .	56
13.37	project_to_curve . . . . .	56
13.38	quad_isconvex . . . . .	56
13.39	random_disk . . . . .	56
13.40	random_simplex . . . . .	57
13.41	sphere_volume . . . . .	57
13.42	tetra_volume . . . . .	57
13.43	tobarycentric . . . . .	57
13.44	tobarycentric1 . . . . .	57
13.45	tobarycentric2 . . . . .	57
13.46	tobarycentric3 . . . . .	57
13.47	tri_angle . . . . .	57
13.48	tri_area . . . . .	58
13.49	tri_centroid . . . . .	58
13.50	tri_distance_opposit_midpoint . . . . .	58
13.51	tri_edge_length . . . . .	58
13.52	tri_edge_midpoint . . . . .	58
13.53	tri_excircle . . . . .	58
13.54	tri_height . . . . .	58
13.55	tri_incircle . . . . .	58
13.56	tri_isacute . . . . .	59
13.57	tri_isobtuse . . . . .	59
13.58	tri_semiperimeter . . . . .	59
13.59	tri_side_length . . . . .	59
<b>14</b>	<b>geometry</b>	<b>59</b>
14.1	Polygon . . . . .	59
14.2	bounding_box . . . . .	59

14.3	curvature_1d . . . . .	60
14.4	cvt . . . . .	60
14.5	deg_to_frac . . . . .	60
14.6	ellipse . . . . .	60
14.7	ellipseX . . . . .	60
14.8	ellipseY . . . . .	60
14.9	first_intersect . . . . .	60
14.10	golden_ratio . . . . .	60
14.11	hypot3 . . . . .	61
14.12	meanangle . . . . .	61
14.13	meanangle2 . . . . .	61
14.14	meanangle3 . . . . .	61
14.15	meanangle4 . . . . .	61
14.16	medianangle . . . . .	61
14.17	medianangle2 . . . . .	61
14.18	pilim . . . . .	62
14.19	streamline_radius_of_curvature . . . . .	62
<b>15 histogram/@Histogram</b>		<b>62</b>
15.1	2x . . . . .	62
15.2	Histogram . . . . .	62
15.3	bimodes . . . . .	62
15.4	cdf . . . . .	62
15.5	cdfS . . . . .	62
15.6	chi2test . . . . .	62
15.7	cmoment . . . . .	63
15.8	cmomentS . . . . .	63
15.9	entropy . . . . .	63
15.10	entropyS . . . . .	63
15.11	iquantile . . . . .	63
15.12	kstest . . . . .	63
15.13	kurtosis . . . . .	63
15.14	kurtosisS . . . . .	63
15.15	mean . . . . .	63
15.16	meanS . . . . .	63
15.17	median . . . . .	64
15.18	medianS . . . . .	64
15.19	mode . . . . .	64
15.20	modeS . . . . .	64
15.21	moment . . . . .	64
15.22	momentS . . . . .	64
15.23	pdf . . . . .	64
15.24	quantile . . . . .	64
15.25	quantileS . . . . .	64

15.26	setup . . . . .	64
15.27	skewness . . . . .	65
15.28	skewnessS . . . . .	65
15.29	stairs . . . . .	65
15.30	stairsS . . . . .	65
15.31	std . . . . .	65
15.32	stdS . . . . .	65
15.33	var . . . . .	65
15.34	varS . . . . .	65
<b>16</b>	<b>histogram</b>	<b>65</b>
16.1	hist_man . . . . .	65
16.2	histadapt . . . . .	66
16.3	histconst . . . . .	66
16.4	pdf_poly . . . . .	66
16.5	plotcdf . . . . .	66
16.6	test_histogram . . . . .	66
<b>17</b>	<b>linear-algebra</b>	<b>66</b>
17.1	averaging_matrix_2 . . . . .	66
17.2	colnorm . . . . .	66
17.3	condest_ . . . . .	66
<b>18</b>	<b>linear-algebra/coordinate-transformation</b>	<b>67</b>
18.1	barycentric2cartesian . . . . .	67
18.2	barycentric2cartesian3 . . . . .	67
18.3	cartesian2barycentric . . . . .	67
18.4	cartesian_to_unit_triangle_basis . . . . .	67
18.5	ellipsoid2geoid . . . . .	67
18.6	example_approximate_utm_conversion . . . . .	67
18.7	latlon2utm . . . . .	67
18.8	latlon2utm_simple . . . . .	67
18.9	lowrance_mercator_to_wgs84 . . . . .	68
18.10	nmea2utm . . . . .	68
18.11	sn2xy . . . . .	68
18.12	unit_triangle_to_cartesian . . . . .	68
18.13	utm2latlon . . . . .	68
18.14	xy2nt . . . . .	68
18.15	xy2sn . . . . .	68
18.16	xy2sn_java . . . . .	69
18.17	xy2sn_old . . . . .	69
<b>19</b>	<b>linear-algebra</b>	<b>69</b>
19.1	det2x2 . . . . .	69

19.2	det3x3 . . . . .	69
19.3	det4x4 . . . . .	69
19.4	diag2x2 . . . . .	69
19.5	eig2x2 . . . . .	69
<b>20</b>	<b>linear-algebra/eigenvalue</b>	<b>70</b>
20.1	eig_bisection . . . . .	70
20.2	eig_inverse . . . . .	70
20.3	eig_inverse_iteration . . . . .	70
20.4	eig_power_iteration . . . . .	70
<b>21</b>	<b>linear-algebra/eigenvalue/jacobi-davidson</b>	<b>70</b>
21.1	afun_jdm . . . . .	70
21.2	davidson . . . . .	70
21.3	jacobi_davidson . . . . .	70
21.4	jacobi_davidson_qr . . . . .	70
21.5	jacobi_davidson_qz . . . . .	70
21.6	jacobi_davidson_simple . . . . .	71
21.7	jdqr . . . . .	71
21.8	jdqr_sleijpen . . . . .	74
21.9	jdqr_vorst . . . . .	78
21.10	jdqz . . . . .	80
21.11	mfunc_jdm . . . . .	85
21.12	mgs . . . . .	85
21.13	minres_ . . . . .	85
21.14	mv_jacobi_davidson . . . . .	85
<b>22</b>	<b>linear-algebra</b>	<b>85</b>
22.1	first . . . . .	85
22.2	gershgorin_circle . . . . .	85
22.3	haussdorff . . . . .	86
22.4	ieig2x2 . . . . .	86
22.5	inv2x2 . . . . .	86
22.6	inv3x3 . . . . .	86
22.7	inv4x4 . . . . .	86
<b>23</b>	<b>linear-algebra/lanczos</b>	<b>86</b>
23.1	arnoldi . . . . .	86
23.2	arnoldi_new . . . . .	86
23.3	eigs_lanczos_man . . . . .	87
23.4	lanczos . . . . .	87
23.5	lanczos_ . . . . .	87
23.6	lanczos_biorthogonal . . . . .	87
23.7	lanczos_biorthogonal_improved . . . . .	87



23.8	lanczos_ghep . . . . .	87
23.9	mv_lanczos . . . . .	87
23.10	reorthogonalise . . . . .	87
23.11	test_lanczos . . . . .	87
<b>24</b>	<b>linear-algebra/linear-systems</b>	<b>88</b>
24.1	gmres_man . . . . .	88
24.2	minres_recycle . . . . .	88
<b>25</b>	<b>linear-algebra</b>	<b>88</b>
25.1	lpmean . . . . .	88
25.2	lpnorm . . . . .	88
25.3	matvec3 . . . . .	88
25.4	max2d . . . . .	88
25.5	mid . . . . .	88
25.6	mpoweri . . . . .	89
25.7	mtimes2x2 . . . . .	89
25.8	mtimes3x3 . . . . .	89
25.9	nannorm . . . . .	89
25.10	nanshift . . . . .	89
25.11	nl . . . . .	89
25.12	normalise . . . . .	89
25.13	normalize1 . . . . .	90
25.14	normrows . . . . .	90
25.15	orth2 . . . . .	90
25.16	orth_man . . . . .	90
25.17	orthogonalise . . . . .	90
25.18	paddext . . . . .	90
25.19	paddval1 . . . . .	90
25.20	paddval2 . . . . .	90
<b>26</b>	<b>linear-algebra/polynomial</b>	<b>91</b>
26.1	chebychev . . . . .	91
26.2	piecewise_polynomial . . . . .	91
26.3	roots1 . . . . .	91
26.4	roots2 . . . . .	91
26.5	roots2poly . . . . .	91
26.6	roots3 . . . . .	91
26.7	roots4 . . . . .	91
26.8	roots_piecewise_linear . . . . .	91
26.9	test_roots4 . . . . .	92
26.10	vanderi_1d . . . . .	92
<b>27</b>	<b>linear-algebra</b>	<b>92</b>

27.1	randrot . . . . .	92
27.2	right . . . . .	92
27.3	rot2 . . . . .	92
27.4	rot2dir . . . . .	92
27.5	rot3 . . . . .	92
27.6	rotR . . . . .	92
27.7	rownorm . . . . .	93
27.8	similarity_matrix . . . . .	93
27.9	spnorm . . . . .	93
27.10	spzeros . . . . .	93
27.11	test_roots3 . . . . .	93
27.12	transform_minmax . . . . .	93
27.13	transpose3 . . . . .	93
27.14	transposeall . . . . .	93
<b>28</b>	<b>logic</b>	<b>94</b>
28.1	bitor_man . . . . .	94
<b>29</b>	<b>master/plot</b>	<b>94</b>
29.1	attach_boundary_value . . . . .	94
29.2	cartesian_polar . . . . .	94
29.3	img_vargrid . . . . .	94
29.4	plot_basis_functions . . . . .	94
29.5	plot_convergence . . . . .	94
29.6	plot_dof . . . . .	94
29.7	plot_eigenbar . . . . .	94
29.8	plot_error_estimation . . . . .	95
29.9	plot_error_estimation_2 . . . . .	95
29.10	plot_error_fem . . . . .	95
29.11	plot_fdm_kernel . . . . .	95
29.12	plot_fdm_vs_fem . . . . .	95
29.13	plot_fem_accuracy . . . . .	95
29.14	plot_function_and_grid . . . . .	95
29.15	plot_hat . . . . .	95
29.16	plot_hydrogen_wf . . . . .	95
29.17	plot_mesh . . . . .	95
29.18	plot_mesh_2 . . . . .	96
29.19	plot_refine . . . . .	96
29.20	plot_refine_3d . . . . .	96
29.21	plot_runtime . . . . .	96
29.22	plot_spectrum . . . . .	96
29.23	plot_wavefunction . . . . .	96
<b>30</b>	<b>master/ported</b>	<b>96</b>

30.1	assemble_2d_phi_phi . . . . .	96
30.2	assemble_3d_dphi_dphi . . . . .	96
30.3	assemble_3d_phi_phi . . . . .	96
30.4	dV_2d_ . . . . .	97
30.5	derivative_2d . . . . .	97
30.6	derivative_3d . . . . .	97
30.7	element_neighbour_2d . . . . .	97
30.8	prefetch_2d_ . . . . .	97
30.9	promote_2d_3_10 . . . . .	97
30.10	promote_2d_3_15 . . . . .	97
30.11	promote_2d_3_21 . . . . .	97
30.12	promote_2d_3_6 . . . . .	97
30.13	promote_3d_4_10 . . . . .	97
30.14	promote_3d_4_20 . . . . .	98
30.15	promote_3d_4_35 . . . . .	98
30.16	vander_2d . . . . .	98
30.17	vander_3d . . . . .	98
<b>31</b>	<b>number-theory</b>	<b>98</b>
31.1	ceiln . . . . .	98
31.2	digitsb . . . . .	98
31.3	floorn . . . . .	98
31.4	iseven . . . . .	98
31.5	multichoosek . . . . .	99
31.6	nchoosek_man . . . . .	99
31.7	pythagorean_triple . . . . .	99
31.8	roundn . . . . .	99
<b>32</b>	<b>numerical-methods/differentiation</b>	<b>99</b>
32.1	derivative1 . . . . .	99
32.2	derivative2 . . . . .	99
<b>33</b>	<b>numerical-methods/finite-difference</b>	<b>100</b>
33.1	cdiff . . . . .	100
33.2	cdiffb . . . . .	100
33.3	central_difference . . . . .	100
33.4	cmean . . . . .	100
33.5	cmean2 . . . . .	100
33.6	derivative_matrix_1_1d . . . . .	100
33.7	derivative_matrix_2_1d . . . . .	100
33.8	derivative_matrix_2d . . . . .	101
33.9	derivative_matrix_curvilinear . . . . .	101
33.10	derivative_matrix_curvilinear_2 . . . . .	101
33.11	difference_kernel . . . . .	101

33.12	distmat . . . . .	101
33.13	downwind_difference . . . . .	101
33.14	gradpde2d . . . . .	101
33.15	laplacian . . . . .	102
33.16	laplacian_fdm . . . . .	102
33.17	left . . . . .	102
33.18	lrmean . . . . .	102
<b>34</b>	<b>numerical-methods/finite-difference/master</b>	<b>102</b>
34.1	fdm_adaptive_grid . . . . .	102
34.2	fdm_adaptive_refinement_old . . . . .	102
34.3	fdm_assemble_d1_2d . . . . .	102
34.4	fdm_assemble_d2_2d . . . . .	102
34.5	fdm_confinement . . . . .	103
34.6	fdm_d_vargrid . . . . .	103
34.7	fdm_h_unstructured . . . . .	103
34.8	fdm_hydrogen_vargrid . . . . .	103
34.9	fdm_mark_unstructured_2d . . . . .	103
34.10	fdm_plot . . . . .	103
34.11	fdm_plot_series . . . . .	103
34.12	fdm_refine_2d . . . . .	103
34.13	fdm_refine_3d . . . . .	103
34.14	fdm_refine_unstructured_2d . . . . .	103
34.15	fdm_schroedinger_2d . . . . .	104
34.16	fdm_schroedinger_3d . . . . .	104
34.17	relocate . . . . .	104
<b>35</b>	<b>numerical-methods/finite-difference</b>	<b>104</b>
35.1	mid . . . . .	104
35.2	pwmid . . . . .	104
35.3	ratio . . . . .	104
35.4	steplength . . . . .	104
35.5	swapoddeven . . . . .	104
35.6	test_derivative_matrix_2d . . . . .	105
35.7	test_derivative_matrix_curvilinear . . . . .	105
35.8	test_difference_kernel . . . . .	105
35.9	upwind_difference . . . . .	105
<b>36</b>	<b>numerical-methods/finite-element</b>	<b>105</b>
36.1	Mesh_2d_java . . . . .	105
36.2	Tree_2d_java . . . . .	105
36.3	assemble_1d_dphi_dphi . . . . .	105
36.4	assemble_1d_phi_phi . . . . .	105
36.5	assemble_2d_dphi_dphi_java . . . . .	105

36.6	assemble_2d_phi_phi_java . . . . .	106
36.7	assemble_3d_dphi_dphi_java . . . . .	106
36.8	assemble_3d_phi_phi_java . . . . .	106
36.9	boundary_1d . . . . .	106
36.10	boundary_2d . . . . .	106
36.11	boundary_3d . . . . .	106
36.12	check_area_2d . . . . .	106
36.13	circmesh . . . . .	106
36.14	cropradius . . . . .	106
36.15	display_2d . . . . .	106
36.16	display_3d . . . . .	107
36.17	distort . . . . .	107
36.18	err_2d . . . . .	107
36.19	estimate_err_2d_3 . . . . .	107
36.20	example_1d . . . . .	107
36.21	example_2d . . . . .	107
36.22	explode . . . . .	107
36.23	fem_2d . . . . .	107
36.24	fem_2d_heuristic_mesh . . . . .	107
36.25	fem_get_2d_radial . . . . .	107
36.26	fem_interpolation . . . . .	108
36.27	fem_plot_1d . . . . .	108
36.28	fem_plot_1d_series . . . . .	108
36.29	fem_plot_2d . . . . .	108
36.30	fem_plot_2d_series . . . . .	108
36.31	fem_plot_3d . . . . .	108
36.32	fem_plot_3d_series . . . . .	108
36.33	fem_plot_confine_series . . . . .	108
36.34	fem_radial . . . . .	108
36.35	flip_2d . . . . .	109
36.36	get_mesh_arrays . . . . .	109
36.37	hashkey . . . . .	109
<b>37</b>	<b>numerical-methods/finite-element/int</b>	<b>109</b>
37.1	int_1d_gauss . . . . .	109
37.2	int_1d_gauss_1 . . . . .	109
37.3	int_1d_gauss_2 . . . . .	109
37.4	int_1d_gauss_3 . . . . .	109
37.5	int_1d_gauss_4 . . . . .	109
37.6	int_1d_gauss_5 . . . . .	109
37.7	int_1d_gauss_6 . . . . .	110
37.8	int_1d_gauss_lobatto . . . . .	110
37.9	int_1d_gauss_n . . . . .	110
37.10	int_1d_nc_2 . . . . .	110

37.11	int_1d_nc_3 . . . . .	110
37.12	int_1d_nc_4 . . . . .	110
37.13	int_1d_nc_5 . . . . .	110
37.14	int_1d_nc_6 . . . . .	110
37.15	int_1d_nc_7 . . . . .	110
37.16	int_1d_nc_7_hardy . . . . .	110
37.17	int_2d_gauss_1 . . . . .	111
37.18	int_2d_gauss_12 . . . . .	111
37.19	int_2d_gauss_13 . . . . .	111
37.20	int_2d_gauss_16 . . . . .	111
37.21	int_2d_gauss_19 . . . . .	111
37.22	int_2d_gauss_25 . . . . .	111
37.23	int_2d_gauss_3 . . . . .	111
37.24	int_2d_gauss_33 . . . . .	111
37.25	int_2d_gauss_4 . . . . .	111
37.26	int_2d_gauss_6 . . . . .	111
37.27	int_2d_gauss_7 . . . . .	112
37.28	int_2d_gauss_9 . . . . .	112
37.29	int_2d_nc_10 . . . . .	112
37.30	int_2d_nc_15 . . . . .	112
37.31	int_2d_nc_21 . . . . .	112
37.32	int_2d_nc_3 . . . . .	112
37.33	int_2d_nc_6 . . . . .	112
37.34	int_3d_gauss_1 . . . . .	112
37.35	int_3d_gauss_11 . . . . .	112
37.36	int_3d_gauss_14 . . . . .	112
37.37	int_3d_gauss_15 . . . . .	113
37.38	int_3d_gauss_24 . . . . .	113
37.39	int_3d_gauss_4 . . . . .	113
37.40	int_3d_gauss_45 . . . . .	113
37.41	int_3d_gauss_5 . . . . .	113
37.42	int_3d_nc_11 . . . . .	113
37.43	int_3d_nc_4 . . . . .	113
37.44	int_3d_nc_6 . . . . .	113
37.45	int_3d_nc_8 . . . . .	113

### **38 numerical-methods/finite-element 114**

38.1	interpolation_matrix . . . . .	114
38.2	mark . . . . .	114
38.3	mark_1d . . . . .	114
38.4	mesh_1d_uniform . . . . .	114
38.5	mesh_3d_uniform . . . . .	114
38.6	mesh_interpolate . . . . .	114
38.7	neighbour_1d . . . . .	114

38.8	old	114
38.9	pdeeig_1d	114
38.10	pdeeig_2d	115
38.11	pdeeig_3d	115
38.12	polynomial_derivative_1d	115
38.13	potential_const	115
38.14	potential_coulomb	115
38.15	potential_harmonic_oscillator	115
38.16	project_circle	115
38.17	project_rectangle	115
38.18	promote_1d_2.3	115
38.19	promote_1d_2.4	115
38.20	promote_1d_2.5	116
38.21	promote_1d_2.6	116
38.22	quadrilateate	116
38.23	recalculate_regularity_2d	116
38.24	refine_1d	116
38.25	refine_2d_21	116
38.26	refine_2d_structural	116
38.27	regularity_1d	116
38.28	regularity_2d	116
38.29	regularity_3d	117
38.30	relocate_2d	117
38.31	test_circmesh	117
38.32	test_hermite	117
38.33	tri_assign_points	117
38.34	triangulation_uniform	117
38.35	vander_1d	117
38.36	vanderd_1d	117
38.37	vanderi_1d	117
<b>39</b>	<b>numerical-methods/finite-volume/@Advection</b>	<b>118</b>
39.1	Advection	118
39.2	dot_advection	118
<b>40</b>	<b>numerical-methods/finite-volume/@Burgers</b>	<b>118</b>
40.1	burgers_split	118
40.2	dot_burgers_fdm	118
40.3	dot_burgers_fft	118
<b>41</b>	<b>numerical-methods/finite-volume/@Finite_Volume</b>	<b>118</b>
41.1	Finite_Volume	118
41.2	apply_bc	119
41.3	solve	119

41.4	step_split_strang . . . . .	119
41.5	step_unsplit . . . . .	119
<b>42</b>	<b>numerical-methods/finite-volume/@Flux_Limiter</b>	<b>119</b>
42.1	Flux_Limiter . . . . .	119
42.2	beam_warming . . . . .	119
42.3	fromm . . . . .	120
42.4	lax_wendroff . . . . .	120
42.5	minmod . . . . .	120
42.6	monotized_central . . . . .	120
42.7	muscl . . . . .	120
42.8	superbee . . . . .	120
42.9	upwind . . . . .	120
42.10	vanLeer . . . . .	121
<b>43</b>	<b>numerical-methods/finite-volume/@KDV</b>	<b>121</b>
43.1	dot_kdv_fdm . . . . .	121
43.2	dot_kdv_fft . . . . .	121
43.3	kdv_split . . . . .	121
<b>44</b>	<b>numerical-methods/finite-volume/@Reconstruct_Average_Evolve</b>	<b>121</b>
44.1	Reconstruct_Average_Evolve . . . . .	121
44.2	advect_highres . . . . .	122
44.3	advect_lowress . . . . .	122
<b>45</b>	<b>numerical-methods/finite-volume</b>	<b>122</b>
45.1	Godunov . . . . .	122
45.2	Lax_Friedrich . . . . .	122
45.3	Measure . . . . .	122
45.4	Roe . . . . .	122
45.5	fv_swe . . . . .	123
45.6	staggered_euler . . . . .	123
45.7	staggered_grid . . . . .	123
<b>46</b>	<b>numerical-methods</b>	<b>123</b>
46.1	grid2quad . . . . .	123
<b>47</b>	<b>numerical-methods/integration</b>	<b>123</b>
47.1	cumintL . . . . .	123
47.2	cumintR . . . . .	123
47.3	int_trapezoidal . . . . .	123
<b>48</b>	<b>numerical-methods/interpolation/@Kriging</b>	<b>124</b>
48.1	Kriging . . . . .	124
48.2	estimate_semivariance . . . . .	124



48.3	interpolate_ . . . . .	124
<b>49</b>	<b>numerical-methods/interpolation/@RegularizedInterpolator1</b>	<b>124</b>
49.1	RegularizedInterpolator1 . . . . .	124
49.2	init . . . . .	124
<b>50</b>	<b>numerical-methods/interpolation/@RegularizedInterpolator2</b>	<b>125</b>
50.1	RegularizedInterpolator2 . . . . .	125
50.2	init . . . . .	125
<b>51</b>	<b>numerical-methods/interpolation/@RegularizedInterpolator3</b>	<b>125</b>
51.1	RegularizedInterpolator3 . . . . .	125
51.2	init . . . . .	125
<b>52</b>	<b>numerical-methods/interpolation</b>	<b>125</b>
52.1	IDW . . . . .	125
52.2	IPoly . . . . .	125
52.3	IRBM . . . . .	126
52.4	ISparse . . . . .	126
52.5	Inn . . . . .	126
52.6	Interpolator . . . . .	126
52.7	fixnan . . . . .	126
52.8	idw1 . . . . .	126
52.9	idw2 . . . . .	126
52.10	inner2outer . . . . .	127
52.11	inner2outer2 . . . . .	127
52.12	interp1_limited . . . . .	127
52.13	interp1_man . . . . .	127
52.14	interp1_piecewise_linear . . . . .	127
52.15	interp1_save . . . . .	127
52.16	interp1_slope . . . . .	128
52.17	interp1_smooth . . . . .	128
52.18	interp1_unique . . . . .	128
52.19	interp2_man . . . . .	128
52.20	interp_angle . . . . .	128
52.21	interp_fourier . . . . .	128
52.22	interp_fourier_batch . . . . .	128
52.23	interp_sn . . . . .	129
52.24	interp_sn2 . . . . .	129
52.25	interp_sn3 . . . . .	129
52.26	interp_sn_ . . . . .	129
52.27	limit_by_distance_1d . . . . .	129
52.28	resample1 . . . . .	129
52.29	resample_d_min . . . . .	129

52.30	resample_vector . . . . .	130
52.31	test_interp1_limited . . . . .	130
<b>53</b>	<b>numerical-methods</b>	<b>130</b>
53.1	inverse_complex . . . . .	130
53.2	maccormack_step . . . . .	130
<b>54</b>	<b>numerical-methods/ode/@BVPS_Characteristic</b>	<b>130</b>
54.1	BVPS_Characteristic . . . . .	130
54.2	assemble1_A . . . . .	130
54.3	assemble1_A_Q . . . . .	130
54.4	assemble2_A . . . . .	130
54.5	assemble_AA . . . . .	131
54.6	assemble_AAA . . . . .	131
54.7	bvp1c . . . . .	131
54.8	check_arguments . . . . .	131
54.9	couple_junctions . . . . .	131
54.10	derivative . . . . .	131
54.11	init . . . . .	131
54.12	reconstruct . . . . .	131
54.13	resample . . . . .	131
54.14	solve . . . . .	132
<b>55</b>	<b>numerical-methods/ode/@Time_Stepper</b>	<b>132</b>
55.1	Time_Stepper . . . . .	132
55.2	solve . . . . .	132
<b>56</b>	<b>numerical-methods/ode</b>	<b>132</b>
56.1	bvp2fdm . . . . .	132
56.2	bvp2wavetrain . . . . .	133
56.3	bvp2wavetwopass . . . . .	133
56.4	ivp_euler_forward . . . . .	133
56.5	ivprk2 . . . . .	133
56.6	ode2_matrix . . . . .	133
56.7	ode2characteristic . . . . .	133
56.8	step_trapezoidal . . . . .	134
56.9	test_bvp2 . . . . .	134
<b>57</b>	<b>numerical-methods/optimisation</b>	<b>134</b>
57.1	armijo_stopping_criterion . . . . .	134
57.2	astar . . . . .	134
57.3	binsearch . . . . .	134
57.4	bisection . . . . .	134
57.5	box1 . . . . .	134

57.6	box2 . . . . .	134
57.7	cauchy . . . . .	135
57.8	cauchy2 . . . . .	135
57.9	directional_derivative . . . . .	135
57.10	dud . . . . .	135
57.11	extreme3 . . . . .	135
57.12	extreme_quadratic . . . . .	136
57.13	ftest . . . . .	136
57.14	fzero_bisect . . . . .	136
57.15	fzero_newton . . . . .	136
57.16	grad . . . . .	136
57.17	hessian . . . . .	136
57.18	hessian_from_gradient . . . . .	136
57.19	hessian_projected . . . . .	136
57.20	line_search . . . . .	136
57.21	line_search2 . . . . .	137
57.22	line_search_polynomial . . . . .	137
57.23	line_search_polynomial2 . . . . .	137
57.24	line_search_quadratic . . . . .	137
57.25	line_search_quadratic2 . . . . .	138
57.26	line_search_wolfe . . . . .	138
57.27	ls_bgfs . . . . .	138
57.28	ls_broyden . . . . .	138
57.29	ls_generalized_secant . . . . .	139
57.30	nlcg . . . . .	139
57.31	nlls . . . . .	139
57.32	picard . . . . .	139
57.33	poly_extrema . . . . .	139
57.34	quadratic_function . . . . .	139
57.35	quadratic_programming . . . . .	139
57.36	quadratic_step . . . . .	140
57.37	rosenbrock . . . . .	140
57.38	sqrt_heron . . . . .	140
57.39	test_directional_derivative . . . . .	140
57.40	test_dud . . . . .	140
57.41	test_fzero_newton . . . . .	140
57.42	test_line_search_quadratic2 . . . . .	140
57.43	test_ls_generalized_secant . . . . .	140
57.44	test_nlcg_6_order . . . . .	140
57.45	test_nlls . . . . .	141
<b>58</b>	<b>numerical-methods/pde</b>	<b>141</b>
58.1	laplacian2d_fundamental_solution . . . . .	141

<b>59</b>	<b>numerical-methods/piecewise-polynomials</b>	<b>141</b>
59.1	Hermite1 . . . . .	141
59.2	hp2_fit . . . . .	141
59.3	hp2_predict . . . . .	141
59.4	hp_predict . . . . .	141
59.5	hp_regress . . . . .	142
59.6	lp_count . . . . .	142
59.7	lp_predict . . . . .	142
59.8	lp_regress . . . . .	142
59.9	lp_regress_ . . . . .	142
<b>60</b>	<b>numerical-methods</b>	<b>142</b>
60.1	test_adams_bashforth . . . . .	142
<b>61</b>	<b>regression/@PolyOLS</b>	<b>142</b>
61.1	PolyOLS . . . . .	142
61.2	coefftest . . . . .	142
61.3	detrend . . . . .	143
61.4	fit . . . . .	143
61.5	fit_ . . . . .	143
61.6	predict . . . . .	143
61.7	predict_ . . . . .	143
61.8	slope . . . . .	143
<b>62</b>	<b>regression/@PowerLS</b>	<b>143</b>
62.1	PowerLS . . . . .	143
62.2	fit . . . . .	144
62.3	predict . . . . .	144
62.4	predict_ . . . . .	144
<b>63</b>	<b>regression/@Theil</b>	<b>144</b>
63.1	Theil . . . . .	144
63.2	detrend . . . . .	144
63.3	fit . . . . .	144
63.4	predict . . . . .	145
63.5	slope . . . . .	145
<b>64</b>	<b>regression</b>	<b>145</b>
64.1	Theil_Multivariate . . . . .	145
64.2	areg . . . . .	145
64.3	ginireg . . . . .	145
64.4	hesssimplereg . . . . .	145
64.5	l1lin . . . . .	145
64.6	lsq_sparam . . . . .	146

64.7	polyfitd . . . . .	146
64.8	regression_method_of_moments . . . . .	146
64.9	robustlinreg . . . . .	146
64.10	theil2 . . . . .	146
64.11	theil_generalised . . . . .	146
64.12	total_least_squares . . . . .	147
64.13	weighted_median_regression . . . . .	147
<b>65</b>	<b>set-theory</b>	<b>147</b>
65.1	issubset . . . . .	147
<b>66</b>	<b>signal-processing</b>	<b>147</b>
66.1	acf_effective_sample_size . . . . .	147
66.2	acf_genton . . . . .	147
66.3	acfar1 . . . . .	148
66.4	acfar1_2 . . . . .	148
66.5	acfar2 . . . . .	148
66.6	acfar2_2 . . . . .	148
66.7	ar1_cutoff_frequency . . . . .	148
66.8	ar1_effective_sample_size . . . . .	148
66.9	ar1_mse_mu_single_sample . . . . .	148
66.10	ar1_mse_pop . . . . .	149
66.11	ar1_mse_range . . . . .	149
66.12	ar1_spectrum . . . . .	149
66.13	ar1_to_tikhonov . . . . .	149
66.14	ar1_var_factor . . . . .	149
66.15	ar1_var_factor_ . . . . .	149
66.16	ar1_var_range2 . . . . .	149
66.17	ar1delay . . . . .	150
66.18	ar1delay_old . . . . .	150
66.19	ar2conv . . . . .	150
66.20	ar2dof . . . . .	150
66.21	ar2param . . . . .	150
66.22	asymwin . . . . .	150
66.23	autocorr_fft . . . . .	150
66.24	bandpass . . . . .	151
66.25	bandpass2 . . . . .	151
66.26	bartlett . . . . .	151
66.27	bartlett_spectrogram . . . . .	151
66.28	bin1d . . . . .	151
66.29	bin2d . . . . .	151
66.30	binormrnd . . . . .	152
66.31	conv1_man . . . . .	152
66.32	conv2_man . . . . .	152

66.33	conv2z . . . . .	152
66.34	conv30 . . . . .	152
66.35	conv_ . . . . .	152
66.36	conv_centered . . . . .	152
66.37	convz . . . . .	152
66.38	cosexpdelay . . . . .	153
66.39	csmooth . . . . .	153
66.40	daniell_window . . . . .	153
66.41	danielle_window . . . . .	153
66.42	db2neper . . . . .	153
66.43	db2power . . . . .	153
66.44	derive_danielle_weight . . . . .	153
66.45	derive_limit_0_acfar . . . . .	153
66.46	detect_peak . . . . .	154
66.47	digital_low_pass_filter . . . . .	154
66.48	doublesum_ij . . . . .	154
66.49	effective_sample_size_to_ar1 . . . . .	154
66.50	filt_hodges_lehman . . . . .	154
66.51	filter1 . . . . .	154
66.52	filter2 . . . . .	154
66.53	filter_ . . . . .	155
66.54	filteriir . . . . .	155
66.55	filterp . . . . .	155
66.56	filterp1 . . . . .	156
66.57	filterstd . . . . .	156
66.58	firls_man . . . . .	156
66.59	flattopwin . . . . .	156
66.60	frequency_response_boxcar . . . . .	156
66.61	freqz_boxcar . . . . .	156
66.62	gaussfilt1 . . . . .	156
66.63	hanchangewin . . . . .	156
66.64	hanchangewin2 . . . . .	157
66.65	hanwin . . . . .	157
66.66	hanwin_ . . . . .	157
66.67	highpass . . . . .	157
66.68	kaiserwin . . . . .	157
66.69	kalman . . . . .	157
66.70	lanczoswin . . . . .	157
66.71	last . . . . .	157
66.72	lowpass . . . . .	158
66.73	lowpass2 . . . . .	158
66.74	lowpass_iir . . . . .	158
66.75	lowpass_iir_symmetric . . . . .	158
66.76	lowpassfilter2 . . . . .	158

66.77	maxfilt1 . . . . .	158
66.78	meanfilt1 . . . . .	158
66.79	medfilt1_man . . . . .	158
66.80	medfilt1_man2 . . . . .	159
66.81	medfilt1_padded . . . . .	159
66.82	medfilt1_reduced . . . . .	159
66.83	mid_term_single_sample . . . . .	159
66.84	minfilt1 . . . . .	159
66.85	mu2ar1 . . . . .	159
66.86	mysmooth . . . . .	159
66.87	nanautocorr . . . . .	159
66.88	nanmedfilt1 . . . . .	160
66.89	neper2db . . . . .	160
66.90	peaks_man . . . . .	160
66.91	polyfilt1 . . . . .	160
66.92	qmedfilt1 . . . . .	160
66.93	randar1 . . . . .	160
66.94	randar1_dual . . . . .	160
66.95	randar2 . . . . .	160
66.96	randarp . . . . .	161
66.97	range_window . . . . .	161
66.98	rectwin . . . . .	161
66.99	recursive_sum . . . . .	161
66.100	select_range . . . . .	161
66.101	smooth1d_parametric . . . . .	161
66.102	smooth2 . . . . .	161
66.103	smooth_man . . . . .	161
66.104	smooth_parametric . . . . .	162
66.105	smooth_parametric2 . . . . .	162
66.106	smooth_with_splines . . . . .	162
66.107	smoothfft . . . . .	162
66.108	spectrogram . . . . .	162
66.109	std_window . . . . .	162
66.110	sum_i_lag . . . . .	162
66.111	sum_ii . . . . .	162
66.112	sum_ii_ . . . . .	163
66.113	sum_ij . . . . .	163
66.114	sum_ij_ . . . . .	163
66.115	sum_ij_partial_ . . . . .	163
66.116	sum_multivar . . . . .	163
66.117	test_acfar1 . . . . .	163
66.118	test_acfar1_2 . . . . .	163
66.119	test_acfar1_3 . . . . .	163
66.120	test_acfar1_4 . . . . .	163

66.121	test_acfar2 . . . . .	164
66.122	test_ar1_var_factor . . . . .	164
66.123	test_ar1_var_factor_2 . . . . .	164
66.124	test_ar1_var_mu_single_sample . . . . .	164
66.125	test_ar1_var_pop . . . . .	164
66.126	test_ar1_var_pop_1 . . . . .	164
66.127	test_ar1delay . . . . .	164
66.128	test_bivariate_covariance_term . . . . .	164
66.129	test_convexity . . . . .	164
66.130	test_lanczoswin . . . . .	164
66.131	test_madcorr . . . . .	165
66.132	test_randar1 . . . . .	165
66.133	test_randar1_multivariate . . . . .	165
66.134	test_randar2 . . . . .	165
66.135	test_sum_ij . . . . .	165
66.136	test_sum_multivar . . . . .	165
66.137	test_trifilt1 . . . . .	165
66.138	test_wautocorr . . . . .	165
66.139	test_wavelet_transform . . . . .	165
66.140	test_wordfilt . . . . .	165
66.141	test_xar1_mid_term . . . . .	166
66.142	tikhonov_to_ar1 . . . . .	166
66.143	trapwin . . . . .	166
66.144	trifilt1 . . . . .	166
66.145	triwin . . . . .	166
66.146	triwin2 . . . . .	166
66.147	varar1 . . . . .	166
66.148	welch_spectrogram . . . . .	166
66.149	wfilt . . . . .	167
66.150	winbandpass . . . . .	167
66.151	window_make_odd . . . . .	167
66.152	winfilt0 . . . . .	167
66.153	winlength . . . . .	167
66.154	wmeanfilt . . . . .	167
66.155	wmedfilt . . . . .	167
66.156	wordfilt . . . . .	168
66.157	wordfilt_edgeworth . . . . .	168
66.158	xar1 . . . . .	168
66.159	xcorr_man . . . . .	168

<b>67</b>	<b>sorting</b>	<b>168</b>
67.1	sort2 . . . . .	168
67.2	sort2d . . . . .	168



<b>68</b>	<b>special-functions</b>	<b>168</b>
68.1	bessel_sphere . . . . .	168
68.2	digamma_man . . . . .	169
68.3	hankel_sphere . . . . .	169
68.4	hermite . . . . .	169
68.5	legendre_man . . . . .	169
68.6	neumann_sphere . . . . .	169
<b>69</b>	<b>statistics</b>	<b>169</b>
69.1	atan_s2 . . . . .	169
69.2	beta_mode_to_parameter . . . . .	170
69.3	coefficient_of_determination . . . . .	170
69.4	conditional_expectation_normal . . . . .	170
69.5	correlation_confidence_pearson . . . . .	170
<b>70</b>	<b>statistics/distributions</b>	<b>170</b>
70.1	PDF . . . . .	170
70.2	binorm_separation_coefficient . . . . .	170
70.3	binormcdf . . . . .	170
70.4	binormfit . . . . .	170
70.5	binormpdf . . . . .	171
70.6	edgeworth_cdf . . . . .	171
70.7	edgeworth_pdf . . . . .	171
70.8	logn_mode2param . . . . .	171
70.9	logn_param2mode . . . . .	171
70.10	lognpdf_ . . . . .	171
70.11	pdfsample . . . . .	171
70.12	t2cdf . . . . .	172
70.13	t2inv . . . . .	172
<b>71</b>	<b>statistics</b>	<b>172</b>
71.1	example_standard_error_of_sample_quantiles . . . . .	172
71.2	f_var_finite . . . . .	172
71.3	gamma_mode_to_parameter . . . . .	172
71.4	gaussfit3 . . . . .	172
71.5	gaussfit_quantile . . . . .	172
71.6	geoserr . . . . .	172
71.7	geostd . . . . .	173
71.8	hodges_lehmann_correlation . . . . .	173
71.9	hodges_lehmann_dispersion . . . . .	173
<b>72</b>	<b>statistics/information-theory</b>	<b>173</b>
72.1	akaike_information_criterion . . . . .	173
72.2	bayesian_information_criterion . . . . .	174

<b>73</b>	<b>statistics</b>	<b>174</b>
73.1	kurtncdf . . . . .	174
73.2	kurtnpdf . . . . .	174
73.3	kurtosis_bias_corrected . . . . .	174
73.4	limit . . . . .	174
73.5	logfactorial . . . . .	174
73.6	loglogpdf . . . . .	174
73.7	lognfit_quantile . . . . .	174
73.8	logskewcdf . . . . .	175
73.9	logskewpdf . . . . .	175
<b>74</b>	<b>statistics/logu</b>	<b>175</b>
74.1	lambertw_numeric . . . . .	175
74.2	logtrialtcd . . . . .	175
74.3	logtrialtinv . . . . .	175
74.4	logtrialtmean . . . . .	175
74.5	logtrialtpdf . . . . .	176
74.6	logtrialtrnd . . . . .	176
74.7	logtricdf . . . . .	176
74.8	logtriinv . . . . .	176
74.9	logtrimean . . . . .	176
74.10	logtripdf . . . . .	176
74.11	logtrirnd . . . . .	176
74.12	logucdf . . . . .	176
74.13	logucm . . . . .	177
74.14	loguinv . . . . .	177
74.15	logumean . . . . .	177
74.16	logupdf . . . . .	177
74.17	logurnd . . . . .	177
74.18	loguvar . . . . .	177
74.19	medlogu . . . . .	177
74.20	test_logurnd . . . . .	177
74.21	tricdf . . . . .	178
74.22	triinv . . . . .	178
74.23	trime . . . . .	178
74.24	tripdf . . . . .	178
74.25	trirnd . . . . .	178
<b>75</b>	<b>statistics</b>	<b>178</b>
75.1	max_exprnd . . . . .	178
75.2	maxnnormals . . . . .	178
75.3	mean_generalized_gampdf . . . . .	179
75.4	midrange . . . . .	179
75.5	minavg . . . . .	179

75.6	mode_man . . . . .	179
<b>76</b>	<b>statistics/moment-statistics</b>	<b>179</b>
76.1	autocorr_man3 . . . . .	179
76.2	autocorr_man4 . . . . .	179
76.3	autocorr_man5 . . . . .	179
76.4	blockserr . . . . .	180
76.5	comoment . . . . .	180
76.6	corr_man . . . . .	180
76.7	cov_man . . . . .	180
76.8	dof . . . . .	180
76.9	edgeworth_quantile . . . . .	181
76.10	effective_sample_size . . . . .	181
76.11	f_correlation . . . . .	181
76.12	f_finite . . . . .	181
76.13	lmean . . . . .	181
76.14	lmoment . . . . .	181
76.15	maskmean . . . . .	181
76.16	masknanmean . . . . .	182
76.17	mean1 . . . . .	182
76.18	mean_man . . . . .	182
76.19	mse . . . . .	182
76.20	nanautocorr_man1 . . . . .	182
76.21	nanautocorr_man2 . . . . .	182
76.22	nanautocorr_man4 . . . . .	182
76.23	nancorr . . . . .	183
76.24	nancumsum . . . . .	183
76.25	nanlmean . . . . .	183
76.26	nanr2 . . . . .	183
76.27	nanrms . . . . .	183
76.28	nanrmse . . . . .	183
76.29	nanserr . . . . .	183
76.30	nanwmean . . . . .	183
76.31	nanwstd . . . . .	184
76.32	nanwvar . . . . .	184
76.33	nanxcorr . . . . .	184
76.34	pearson . . . . .	184
76.35	pearson_to_kendall . . . . .	184
76.36	pool_samples . . . . .	184
76.37	qmean . . . . .	184
76.38	range_mean . . . . .	184
76.39	rmse_ . . . . .	185
76.40	serr . . . . .	185
76.41	serr1 . . . . .	185

76.42	test_kskew . . . . .	185
76.43	test_qstd_kskew_optimal_p . . . . .	185
76.44	wautocorr . . . . .	185
76.45	wcorr . . . . .	185
76.46	wcov . . . . .	186
76.47	wdof . . . . .	186
76.48	wkurt . . . . .	186
76.49	wmean . . . . .	186
76.50	wrms . . . . .	186
76.51	wserr . . . . .	186
76.52	wskew . . . . .	186
76.53	wstd . . . . .	187
76.54	wvar . . . . .	187
<b>77</b>	<b>statistics</b>	<b>187</b>
77.1	nangeomean . . . . .	187
77.2	nangeostd . . . . .	187
<b>78</b>	<b>statistics/nonparametric-statistics</b>	<b>187</b>
78.1	kernel1d . . . . .	187
78.2	kernel2d . . . . .	187
<b>79</b>	<b>statistics</b>	<b>188</b>
79.1	normmoment . . . . .	188
79.2	normpdf2 . . . . .	188
<b>80</b>	<b>statistics/order-statistics</b>	<b>188</b>
80.1	hodges_lehmann_location . . . . .	188
80.2	kendall . . . . .	188
80.3	kendall_to_pearson . . . . .	188
80.4	mad2sd . . . . .	188
80.5	madcorr . . . . .	189
80.6	median2_holder . . . . .	189
80.7	median_ci . . . . .	189
80.8	median_man . . . . .	189
80.9	mediani . . . . .	189
80.10	nanmadcorr . . . . .	189
80.11	nanwmedian . . . . .	189
80.12	nanwquantile . . . . .	190
80.13	oja_median . . . . .	190
80.14	qkurtosis . . . . .	190
80.15	qmoments . . . . .	190
80.16	qskew . . . . .	190
80.17	qskewq . . . . .	191

80.18	qstdq . . . . .	191
80.19	quantile1_optimisation . . . . .	191
80.20	quantile2_breckling . . . . .	191
80.21	quantile2_chaudhuri . . . . .	191
80.22	quantile2_projected . . . . .	191
80.23	quantile2_projected2 . . . . .	191
80.24	quantile_envelope . . . . .	191
80.25	quantile_regression_simple . . . . .	192
80.26	ranking . . . . .	192
80.27	spatial_median . . . . .	192
80.28	spatial_quantile . . . . .	192
80.29	spatial_quantile2 . . . . .	192
80.30	spatial_quantile3 . . . . .	192
80.31	spatial_rank . . . . .	192
80.32	spatial_sign . . . . .	192
80.33	spatial_signed_rank . . . . .	193
80.34	spearman . . . . .	193
80.35	spearman_rank . . . . .	193
80.36	spearman_to_pearson . . . . .	193
80.37	wmedian . . . . .	193
80.38	wquantile . . . . .	193
<b>81 statistics</b>		<b>193</b>
81.1	qstd . . . . .	193
81.2	quantile_extrap . . . . .	193
<b>82 statistics/random-number-generation</b>		<b>194</b>
82.1	laplacernd . . . . .	194
82.2	randc . . . . .	194
82.3	skewness2param . . . . .	194
82.4	skewpdf_central_moments . . . . .	194
82.5	skewrnd . . . . .	194
82.6	skewrnd2 . . . . .	194
<b>83 statistics</b>		<b>194</b>
83.1	range . . . . .	194
83.2	resample_with_replacement . . . . .	194
<b>84 statistics/resampling-statistics/@Jackknife</b>		<b>195</b>
84.1	Jackknife . . . . .	195
84.2	estimated_STATIC . . . . .	195
84.3	matrix1_STATIC . . . . .	195
84.4	matrix2 . . . . .	195

<b>85</b>	<b>statistics/resampling-statistics</b>	<b>196</b>
85.1	block_jackknife . . . . .	196
85.2	jackknife_moments . . . . .	196
85.3	moving_block_jackknife . . . . .	196
85.4	randblockserr . . . . .	196
85.5	resample . . . . .	196
<b>86</b>	<b>statistics</b>	<b>197</b>
86.1	scale_quantile_sd . . . . .	197
86.2	sd_sample_quantiles . . . . .	197
86.3	skewpdf . . . . .	197
86.4	test_mean_generalized_gampdf . . . . .	197
86.5	trimmed_mean . . . . .	197
86.6	ttest2_man . . . . .	197
86.7	ttest_man . . . . .	198
86.8	ttest_paired . . . . .	198
86.9	wgeomean . . . . .	198
86.10	wgeovar . . . . .	198
86.11	wharmean . . . . .	198
86.12	wharstd . . . . .	198
86.13	wharvar . . . . .	198
<b>87</b>	<b>mathematics</b>	<b>199</b>
87.1	ternary_diagram . . . . .	199
<b>88</b>	<b>test/master</b>	<b>199</b>
88.1	dat_test_lanczos_3d_k_20_n_40 . . . . .	199
88.2	poisson2d_blk . . . . .	199
88.3	qr_implicit_givens_2 . . . . .	199
88.4	spectral_derivative_2d . . . . .	199
88.5	test_2d_eigensolver_hydrogen . . . . .	199
88.6	test_2d_refine . . . . .	199
88.7	test_3d_eigensolver_hydrogen . . . . .	199
88.8	test_FEM . . . . .	200
88.9	test_Mesh_3d . . . . .	200
88.10	test_arnoldi . . . . .	200
88.11	test_arpackc . . . . .	200
88.12	test_assemble . . . . .	200
88.13	test_assembly_performance . . . . .	200
88.14	test_bc_one_sided . . . . .	200
88.15	test_compare_solvers . . . . .	200
88.16	test_complete . . . . .	200
88.17	test_convergence . . . . .	200
88.18	test_convergence_b . . . . .	201

88.19	test_df_2d . . . . .	201
88.20	test_eig_algs . . . . .	201
88.21	test_eig_inverse . . . . .	201
88.22	test_eigs_lanczos . . . . .	201
88.23	test_eigs_lanczos_1 . . . . .	201
88.24	test_eigs_lanczos_2 . . . . .	201
88.25	test_eigs_lanczos_performance . . . . .	201
88.26	test_fdm . . . . .	201
88.27	test_fdm_d_vargrid . . . . .	201
88.28	test_fdm_spectral . . . . .	202
88.29	test_fem . . . . .	202
88.30	test_fem_1d . . . . .	202
88.31	test_fem_1d_higher_order . . . . .	202
88.32	test_fem_2d_adaptive . . . . .	202
88.33	test_fem_2d_higher_order . . . . .	202
88.34	test_fem_3d_higher_order . . . . .	202
88.35	test_fem_3d_refine . . . . .	202
88.36	test_fem_b . . . . .	202
88.37	test_fem_derivative . . . . .	202
88.38	test_fem_quadrature . . . . .	203
88.39	test_final . . . . .	203
88.40	test_fix_substitution . . . . .	203
88.41	test_forward . . . . .	203
88.42	test_get_sparse_arrays . . . . .	203
88.43	test_harmonic_oscillator . . . . .	203
88.44	test_high_order_fdm_periodic_bc . . . . .	203
88.45	test_hydrogen_wf . . . . .	203
88.46	test_ichol . . . . .	203
88.47	test_interpolation . . . . .	203
88.48	test_inverse_problem . . . . .	204
88.49	test_it_vs_exact . . . . .	204
88.50	test_jama . . . . .	204
88.51	test_jd . . . . .	204
88.52	test_jdqz . . . . .	204
88.53	test_lanczos_2 . . . . .	204
88.54	test_lanczos_biorthogonal . . . . .	204
88.55	test_laplacian . . . . .	204
88.56	test_laplacian_non_uniform . . . . .	204
88.57	test_laplacian_simple . . . . .	204
88.58	test_mesh_2d_uniform . . . . .	205
88.59	test_mesh_2d_uniform_2 . . . . .	205
88.60	test_mesh_circle . . . . .	205
88.61	test_mesh_generation . . . . .	205
88.62	test_mesh_interpolate . . . . .	205

88.63	test_mg . . . . .	205
88.64	test_minres_recycle . . . . .	205
88.65	test_multigrid . . . . .	205
88.66	test_nc . . . . .	205
88.67	test_nonuniform_symmetric . . . . .	205
88.68	test_pde . . . . .	206
88.69	test_permutation . . . . .	206
88.70	test_poisson_fem . . . . .	206
88.71	test_polar . . . . .	206
88.72	test_potential . . . . .	206
88.73	test_powers . . . . .	206
88.74	test_precondition . . . . .	206
88.75	test_project_rectangle . . . . .	206
88.76	test_qr . . . . .	206
88.77	test_quantum_well . . . . .	206
88.78	test_radial_adaptive . . . . .	207
88.79	test_radial_confinement . . . . .	207
88.80	test_radial_fixes . . . . .	207
88.81	test_refine_2d . . . . .	207
88.82	test_refine_2d_b . . . . .	207
88.83	test_refine_3d . . . . .	207
88.84	test_refine_structural . . . . .	207
88.85	test_regularisation . . . . .	207
88.86	test_round_off . . . . .	207
88.87	test_schrödinger_potentials . . . . .	207
88.88	test_uniform_mesh . . . . .	208
88.89	test_vargrid . . . . .	208
<b>89</b>	<b>test</b>	<b>208</b>
89.1	test_gaussfit3 . . . . .	208
89.2	test_geoserr . . . . .	208
89.3	test_lognfit_quantile . . . . .	208
89.4	test_max_normal . . . . .	208
89.5	test_mtimes3x3 . . . . .	208
<b>90</b>	<b>mathematics</b>	<b>208</b>
90.1	vanderd_2d . . . . .	208
<b>91</b>	<b>wavelet</b>	<b>209</b>
91.1	continuous_wavelet_transform . . . . .	209
91.2	cwt_man . . . . .	209
91.3	example_wavelets . . . . .	209
91.4	phasewrap . . . . .	209
91.5	test_cwt_man . . . . .	209



91.6	test_phasewrap . . . . .	209
91.7	test_wavelet . . . . .	209
91.8	test_wavelet2 . . . . .	209
91.9	test_wavelet_analysis . . . . .	210
91.10	test_wavelet_reconstruct . . . . .	210
91.11	test_wtc . . . . .	210
91.12	wavelet . . . . .	210
91.13	wavelet_reconstruct . . . . .	210
91.14	wavelet_transform . . . . .	210
<b>92</b>	<b>mathematics</b>	<b>210</b>
92.1	wrapphase . . . . .	210

## 1 calendar

### 1.1 days\_per\_month

### 1.2 isnight

## 2 mathematics

mathematical functions of various kind

### 2.1 cast\_byte\_to\_integer

cast byte to integer

## 3 complex-analysis

operations on complex numbers

### 3.1 complex\_exp\_product\_im\_im

product of the imaginary part of two complex exponentials

the product has two frequency components

```

input :
    c : complex amplitudes
    o : frequencies
output :
    cp : amplitude of the product
    op : frequencies of the product

```

### 3.2 complex\_exp\_product\_im\_re

product of the imaginary part of one and the real part of a second complex exponential

the product has two frequency components

```

input :
    c : complex amplitudes
    o : frequencies
output :
    cp : amplitude of the product
    op : frequencies of the product

```

### 3.3 complex\_exp\_product\_re\_im

the product has two frequency components

product of the imaginary part of one and the real part of a second complex exponential

```

input :
    c : complex amplitudes
    o : frequencies
output :
    cp : amplitude of the product
    op : frequencies of the product

```

### 3.4 complex\_exp\_product\_re\_re

product of the real part of two complex exponentials

$$\text{re}(c_1 \exp(i\omega_1 x)) * \text{re}(c_2 \exp(i\omega_2 x)) = \frac{1}{2} * (\text{real}(c_1 * c_2 * \exp(i * (\omega_1 + \omega_2) * x)) \dots$$

```
+ real(conj(c1)*c2*exp(i*(n2-n1)*o*x)) )
```

the product has two frequency components

```
input :
  c : complex amplitudes
  o : frequencies
output :
  cp : amplitude of the product
  op : frequencies of the product
```

### 3.5 roots

nth-roots of a complex number

```
input:
c : complex number
n : order of root
  n must be rational, to obtain n solutions
  otherwise no finite set of solutions exists

r : roots of the complex number
```

### 3.6 root\_complex

root of a complex number

### 3.7 test\_imroots

## 4 derivation

derivation of several functions by means of symbolic computation

### 4.1 derive\_acfar1

### 4.2 derive\_ar2param

4.3 `derive_arc_length`

4.4 `derive_fourier_power`

4.5 `derive_fourier_power_exp`

4.6 `derive_laplacian_curvilinear`

4.7 `derive_laplacian_fourier_piecewise_linear`

4.8 `derive_logtripdf`

4.9 `derive_smooth1d_parametric`

5 `derivation/master`

5.1 `derive_bc_one_sided`

5.2 `derive_convergence`

5.3 `derive_error_fdm`

5.4 `derive_fdm_poly`

5.5 `derive_fdm_power`

5.6 `derive_fdm_taylor`

5.7 `derive_fdm_vargrid`

5.8 `derive_fem_2d_mass`

5.9 `derive_fem_error_2d`

5.10 `derive_fem_error_3d`

5.11 `derive_fem_sym_2d`

5.12 `derive_grid_constants`

5.13 `derive_interpolation`

5.14 `derive_laplacian`

5.15 `derive_limit`

5.16 `derive_nc_1d`

5.17 `derive_nc_1d_`

5.18 `derive_nc_2d`

5.19 `derive_nonuniform_symmetric`

%

5.20 `derive_richardson`

5.21 `derive_sum`

5.22 `nn`

**5.23**   `test_derive`

**5.24**   `test_derive_fdm_poly`

**5.25**   `test_filter`

**5.26**   `test_vargrid`

## **6**   `derivation`

derivation of several functions by means of symbolic computation

**6.1**   `simplify_atan`

symbolic simplification of the arcus tangent

## **7**   `mathematics`

mathematical functions of various kind

**7.1**   `entropy`

**7.2**   `exp10`

**7.3**   `filter_twosided`

## 8 finance

8.1 `derive_skewrnd_walsh_paramter`

8.2 `gbm_cdf`

8.3 `gbm_fit`

8.4 `gbm_fit_old`

8.5 `gbm_inv`

8.6 `gbm_mean`

8.7 `gbm_median`

8.8 `gbm_pdf`

8.9 `gbm_simulate`



8.10 `gbm_skewness`

8.11 `gbm_std`

8.12 `gbm_transform_time_step`

8.13 `put_price_black_scholes`

8.14 `skewgbm_simulate`

8.15 `skewrnd_walsh`

## 9 `finance/test`

9.1 `test_gbm`

9.2 `test_gbm_pdf`

9.3 `test_skewrnd_walsh`

## 10 `fourier/@STFT`

### 10.1 `STFT`

class for short time fourier transform

Note: the interval `Ti` should be set to at least  $2 \cdot \max(T)$ , as otherwise coefficients

tend to oscillate in the presence of noise

Note: for convenience, the independent variable is labeled as time (`t`),

but the independent variable is arbitrary, so it works likewise in space

### 10.2 `itransform`

inverse of the short time fourier transform

### 10.3 `stft_`

static wrapper for `STFT`

### 10.4 `stftmat`

transformation matrix for the short time fourier transform

### 10.5 `transform`

short time fourier transform

## 11 `fourier`

support and analysis functions both for the discrete (fast) fourier transform (`dft/fft`)

and continuous fourier analysis (fourier series)

### 11.1 `amplitude_from_peak`

amplitude and standard deviation of the amplitude of a frequency  
component  
represented by a peak in the fourier domain  
input :  
h : peak height  
w : peak width at half height  
  
output:  
a : amplitude in real space  
s : standard deviation of the frequency (!)

## 11.2 dftmtx\_man

fourier matrix in matlab style with a limited number of rows,  
columns of higher frequencies are omitted

input :  
n : number of samples  
nr : number of columns

output :  
F : fourier matrix

## 11.3 example\_fourier\_window

## 11.4 fft\_derivative

derivative by fourier transform  
exponential convergence for periodic functions  
results in spurious oscillations for aperiodic functions

input:  
x : data, sampled in equal intervals  
k : order of the derivative

dx : kth-derivative of x

## 11.5 fft\_man

fast fourier transform for complex input data

input:

F : data in real space

output :

F : fourier transformation of F

## 11.6 fftsmooth

smooth the fourier transform and determine upper and lower bound confidence intervals

input :

f :

sfunc : a smoothing function (for example fir convolution with rectangular window)

returns filtered (mean) value and normalized fir window

nf : window length

nsigma : number of standard deviations for confidence intervals

output :

ff : filtered fourier transform

l : lower bound

u : upper bound

## 11.7 fix\_fourier

fill gaps (missing data) by means of fourier extrapolation

fix periodic data series with fourier interpolation

longest gap should not exceed 1/2 of the shortest time span of interest (1/cutoff frequency)

note: this limit equals the position of first side lobe of the ft of a rectangular window with gap length

## 11.8 fourier\_axis

return axis of frequencies and periods for the discrete fourier transform

as computed by fft (matlab-style)

```

input:
X : sample locations (equal interval)
L : length of samples
n : number of samples

output :
f    : frequencies
T    : periods
mask : mask for 1/2 of the fourier transform
      (as both halves are complex conjugates)
N    : frequency id

```

## 11.9 fourier\_cesaro\_correction

### 11.10 fourier\_coefficient\_piecewise\_linear

fourier series coefficients of a piecewise linear function  
(not coefficient of discrete fourier transform)  
function can be discontinuous between intervals  
scales domain length to  $2\pi$

```

input :
l,r : end points of piecewise linear function
lval, rval : values at end points
L : length of domain
n : number of samples/highest frequency

```

```

output :
a, b : coefficients for frequency components

```

### 11.11 fourier\_coefficient\_piecewise\_linear\_1

fourier series coefficients of a piecewise linear function  
(not coefficient of discrete fourier transform)  
function can be discontinuous between intervals  
scales domain length to  $2\pi$

```

input :
X : end points of piecewise linear function
Y : values at end points

```

output :  
ab : coefficients for frequency components

### 11.12 `fourier_coefficient_ramp3`

fourier series coefficient of a ramp

### 11.13 `fourier_coefficient_ramp_pulse`

fourier series coefficient of a ramp pules

### 11.14 `fourier_coefficient_ramp_step`

fourier coefficient of a ramp-step

### 11.15 `fourier_coefficient_square_pulse`

fourier series coefficients of a square pulse

### 11.16 `fourier_cubic_interaction_coefficients`

### 11.17 `fourier_derivative`

coefficients of the derivative of a fourier series  
not of discrete fourier transform (fft)

### 11.18 `fourier_expand`

expand values of fourier series

### 11.19 `fourier_fit`

fit a fourier series to a set of sample points that are not spaced  
in  
equal intervals

### 11.20 `fourier_interpolate`

interpolate samples  $y$  sampled at moments (location)  $t$  to locations  
 $t_i$

### 11.21 `fourier_matrix`

transformation matrix for a continuous fourier series  
(not for the discrete dft/fft)

### 11.22 `fourier_matrix2`

transformation matrix for a continuous fourier series  
(not for the discrete dft/fft)

### 11.23 `fourier_matrix3`

transformation matrix for the continuous fourier transform  
this is a matrix with  $(2n+1)$  real columns

### 11.24 `fourier_matrix_exp`

transformation matrix for a continuous fourier series  
(not for the discrete dft/fft)

### 11.25 `fourier_multiplicative_interaction_coefficients`

### 11.26 `fourier_power`

powers of a continuous fourier series in sin/cos form

$a^p = (u_r + u_1 \sin(\omega t) + u_2 \sin(\omega t + \delta)) ^p$   
phase of first component assumed 0

frequencies higher than  $2\omega$  ignored in input  
frequencies higher than  $3\omega$  not computed

### 11.27 `fourier_power_exp`

powers of the continuous fourier series

$a^p = (u_r + u_1 \sin(\omega t) + u_2 \sin(\omega t + \delta)) ^p$   
phase of first component assumed 0

higher orders than 2 ignored input  
higher order than 3 not computed in output

$y = a_0 + \sum (a_j \sin(j\omega t) + b_j \cos(j\omega t))$   
 $= \text{Real}(\sum_{i=0}^{\infty} c_i \exp(i\omega t)), c_i = a_i + b_i$

NOT the alternative  $\sum_{i=-\infty}^{\infty} \tilde{c}_i$ , tile  $c_j = 1/2 a_j$   
 $+ 1/2i b_j$

### 11.28 `fourier_predict`

expand a continuous fourier series at times  $t$

### 11.29 `fourier_quadratic_interaction_coefficients`

### 11.30 `fourier_range`

approximate range of a continuous Fourier series with 2 components  
 $\text{range}(y) = \max(y) - \min(y)$



### 11.31 `fourier_regress`

fit a continuous fourier series to a set of sample points not  
sampled  
at equal intervals

### 11.32 `fourier_resampled_fit`

fits coefficients of a continuous fourier transform,  
but stores them as resampled values

### 11.33 `fourier_resampled_predict`

interpolates a continuous fourier series that has been stored as  
values  
at their support points

### 11.34 `fourier_signed_square`

coefficients of the fourier series of  $|\cos a + \cos t|$  ( $\cos a + \cos t$ )  
in general  
     $\cos a$  is midrange  
     $\cos t$  is tidal variation  
c.f Dronkers

### 11.35 `fourier_transform`

continuous fourier transformation of y  
(not discrete fourier transformation dft/fft)

input:

    b : data sampled at equal intervals  
    T : length of data in time or space, i.e. position of last  
        sample if  
        position of first sample is 0  
    T\_max : maximum period to include

output :

    A : fourier matrix

```

p : fourier transformation of b
tt : TODO

```

### 11.36 hyperbolic\_fourier\_box

### 11.37 idftmtx\_man

inverse matrix for the discrete fourier transform in matlab style  
with a limited number of columns, thus ignoring higher frequencies  
keep  $2nc+1$  columns (mean and conj-complex pairs of  $nc$  frequencies)

### 11.38 laplace\_2d\_pwlinear

solution to the Laplacian in two dimensions for a finite  
rectangular domain  
with piecewise constant boundary conditions  
linear system with 4 unknowns per frequency component  
these are coefficients of  $s, c, sh, ch$   
 $(pu*(s + c) + qu*(s' + c'))*(shu + chu) = ru$  % upper bc  
 $(pd*(s + c) + qd*(s' + c'))*(shd + chd) = rd$  % lower bc  
 $(sl + cl)*(pl*(shl + chl) + ql*(shl' + chl')) = rl$  % left  
bc  
 $(sr + cr)*(pr*(shr + chr) + qr*(shr' + chr')) = rr$  % right  
bc  
least squares with piecewise integration  
 $[x0, p, q, r]$  piecewise linear polynomials at the boundaries

### 11.39 nanfft

discrete fourier transform of a data series with gaps

### 11.40 peaks

peaks of the power spectrum of a discrete fourier transform  
rule for peaks: there is no higher value left or right of the "peak"  
"

until the signal drops to  $p \cdot y_{\text{peak}}$ ,  $p = 0.5$

works best, when spectrum has been smoothened

input :

- f : frequency
- y : absolute value of fourier transform (power spectrum)
- L : length in space or time of series

output :

- a0 : amplitude
- s0 : standard deviation (error?) of amplitude
- w0 : width of peak
- lambda = wave length (period?)
- pdx : index of peak
- f : frequency (if not given as input)

#### 11.41 roots\_fourier

zeros of continuous fourier series series

$$f = a_0 + \sum_{j=1}^n a_j \cos(j x) + b_j \sin(j x)$$

#### 11.42 spectral\_density

spectral density

#### 11.43 test\_complex\_exp\_product

#### 11.44 test\_fourier\_filter

#### 11.45 test\_idftmtx

## 12 mathematics

mathematical functions of various kind

### 12.1 gaussfit\_quantile

## 13 geometry/@Geometry

### 13.1 Geometry

### 13.2 arclength

arc length of a two dimensional curve

8th order accurate

does not require the segments length to vary smoothly

note: the curve can be considered parametric, e.g.  $x = x(t)$ ,  $y = y(t)$   
and

and  $t = t(s)$ , but the error term contains derivatives of  $t$ ,  
thus a non smooth  $t$  (strongly varying distance between points)  
requires the scaling as done below

### 13.3 arclength\_old

arc length of a two dimensional function

### 13.4 arclength\_old2

arc length of a two dimensional function

### 13.5 base\_point

base point (fusspunkt), i.e. point on a line with shortest distance  
to another point

### 13.6 base\_point\_limited

base point (Fusspunkt) of a point on a line

### 13.7 centroid

centroid of a polygone

### 13.8 cosa\_min\_max

### 13.9 cross2

cross product in two dimensions

### 13.10 curvature

curvature of a function in two dimensions

### 13.11 ddot

sum of squares of cos of inner angles of triangle

### 13.12 distance

euclidan distance between two points

### 13.13 distance2

euclidean distance between two points  
this function requires a and b of equal dimensions, or the least  
the first pair or second pair to be a scalar

### 13.14 dot

dot product

### 13.15 edge\_length

edge length

### 13.16 enclosed\_angle

angle enclosed between two lines

### 13.17 enclosing\_triangle

smallest enclosing equilateral triangle with bottom side parallel to  
X axis

### 13.18 hexagon

coordinates of a hexagon, scaled and rotated

### 13.19 inPolygon

flag points contained in a polygon  
much faster than matlab internal function

### 13.20 inTetra

flag points contained in tetrahedron

### 13.21 inTetra2

flag points contained in tetrahedron

### 13.22 inTriangle

flag points contained in triangle  
function [flag, c] = inTriangle(P1,P2,P3,P0)

### 13.23 intersect

intersect between two lines

### 13.24 lineintersect

intersect of two lines

### 13.25 lineintersect1

intersect of two lines

### 13.26 minimum\_distance\_lines

minimum distance of two lines in three dimensions

### 13.27 mittenpunkt

mittenpunkt of a triangle

### 13.28 nagelpoint

nagelpoint of a triangle

### 13.29 onLine

### 13.30 orthocentre

orthocentre of triangle

### 13.31 plumb\_line

### 13.32 poly\_area

area of a polygon  
function A = poly\_area(x,y)

### 13.33 poly\_edges

edges of a polygon

### 13.34 poly\_set

associate point at arbitrary location with a polygon it is contained  
in  
and assign the value of the polygon to it

### 13.35 poly\_width

width of polygon width holes by surface normals  
holes / islands separated with NaN  
order of points of outer boundary must be cw  
order of points of holes must be ccw  
note that this function does not give the true width for expanding  
sections  
use voronoi polygons for this

### 13.36 polyxpoly

intersections of two polygons



### 13.37 `project_to_curve`

closest point on a curve with respect to a point at distance to the curve

### 13.38 `quad_isconvex`

### 13.39 `random_disk`

draw random points on the unit disk

### 13.40 `random_simplex`

random point inside of a triangle

### 13.41 `sphere_volume`

volume of a sphere

### 13.42 `tetra_volume`

volume of a tetrahedron

### 13.43 `tobarycentric`

cartesian to barycentric coordinates

### 13.44 `tobarycentric1`

cartesian to barycentric coordinates

### 13.45 tobarycentric2

cartesian to barycentric coordinates

### 13.46 tobarycentric3

cartesian to barycentric coordinates

### 13.47 tri\_angle

cos of angles of a triangle

### 13.48 tri\_area

angle of a triangle

### 13.49 tri\_centroid

centroid of a triangle

### 13.50 tri\_distance\_opposit\_midpoint

distance between corner of a triangle and its opposing mid-point

### 13.51 tri\_edge\_length

edge length of a triangle

### 13.52 tri\_edge\_midpoint

mid point of a triangle

### 13.53 tri\_excircle

excircle of a triangle

### 13.54 tri\_height

height of a triangle

### 13.55 tri\_incircle

incircle of a triangle

### 13.56 tri\_isacute

flag acute triangles

### 13.57 tri\_isobtuse

flag obtuse triangles

### 13.58 tri\_semiperimeter

semiperimeter of a triangle

### 13.59 tri\_side\_length

edge length of triangle

## 14 geometry

### 14.1 Polygon

Simple 2D polygon class

Polygon properties:

x - x coordinates of polygon

y - y coordinates of polygon

nnodes - number of nodes in the polygon

Polygon methods:

in - checks whether given points lie inside, on the edge, or  
outside of the polygon

area - returns the area of the polygon

centerline - computes the centerline of the river

iscw - check whether polygon is clockwise

reverse - reverse the order of the polygon

### 14.2 bounding\_box

bounding box of X

### 14.3 curvature\_1d

curvature of a sampled parametric curve in two dimensions

### 14.4 cvt

centroidal voronoi tessellation

### 14.5 deg\_to\_frac

degree, minutes and seconds to fractions

### 14.6 ellipse

n-points on an ellipse

## 14.7 ellipseX

x-coordinates of y-coordinates of an ellipse

## 14.8 ellipseY

## 14.9 first\_intersect

get first intersection between lines in A and B

## 14.10 golden\_ratio

golden ratio

## 14.11 hypot3

hypothenuse in 3D

## 14.12 meanangle

weighted mean of angles

## 14.13 meanangle2

mean angle

## 14.14 meanangle3

mean angle

## 14.15 meanangle4

mean angle

## 14.16 medianangle

median angle  
angle, that has the smallest squared distance to all others

## 14.17 medianangle2

median angle

input  
alpha : x\*m, [rad] angle

output  
ma : 1\*m, [rad] median angle  
sa : 1\*m, [rad] standard error of median angle for uncorrelated  
error

## 14.18 pilim

limit to +- pi

## 14.19 streamline\_radius\_of\_curvature

streamline radius of curvature  
simplifies when rotate to streamwise coordinates to  $R = 1/dv/ds * u$

# 15 histogram/@Histogram

## 15.1 2x

## 15.2 Histogram

## 15.3 bimodes

## 15.4 cdf

## 15.5 cdfS

## 15.6 chi2test

## 15.7 cmoment

## 15.8 cmomentS

## 15.9 entropy

## 15.10 entropyS

## 15.11 iquantile

**15.12   kstest**

**15.13   kurtosis**

**15.14   kurtosisS**

**15.15   mean**

**15.16   meanS**

**15.17   median**

**15.18   medianS**

**15.19   mode**

**15.20   modeS**

**15.21   moment**



**15.22**   **momentS**

**15.23**   **pdf**

**15.24**   **quantile**

**15.25**   **quantileS**

**15.26**   **setup**

**15.27**   **skewness**

**15.28**   **skewnessS**

**15.29**   **stairs**

**15.30**   **stairsS**

**15.31**   **std**

**15.32**   `stdS`

**15.33**   `var`

**15.34**   `varS`

## **16**   `histogram`

**16.1**   `hist_man`

**16.2**   `histadapt`

**16.3**   `histconst`

**16.4**   `pdf_poly`

**16.5**   `plotcdf`

**16.6**   `test_histogram`

## 17 linear-algebra

### 17.1 averaging\_matrix\_2

### 17.2 colnorm

norms of columns

### 17.3 condest\_

estimation of the condition number

## 18 linear-algebra/coordinate-transformation

### 18.1 barycentric2cartesian

barycentric to cartesian coordinates

### 18.2 barycentric2cartesian3

convert barycentric to cartesian coordinates

### 18.3 cartesian2barycentric

cartesian to barycentric coordinates

### 18.4 cartesian\_to\_unit\_triangle\_basis

transform coordinates into unit triangle

### 18.5 ellipsoid2geoid

## 18.6 example\_approximate\_utm\_conversion

## 18.7 latlon2utm

transform latitude and longitude to WGS84 UTM

## 18.8 latlon2utm\_simple

## 18.9 lowrance\_mercator\_to\_wgs84

convert lowrance coordinates to wgs84

based on spreadsheet by D Whitney King and Patty B at Lowrance

## 18.10 nmea2utm

convert nmea messages to utm coordinates

## 18.11 sn2xy

convert sn to xy coordinates

## 18.12 unit\_triangle\_to\_cartesian

transform coordinates in unit triangle to cartesian coordinates

## 18.13 utm2latlon

convert wgs84 utm to latitude and longitude

## 18.14 xy2nt

project all points onto the cross section and assign them nz-coordinates

transform coordinate into N-T reference  
rotate coordinate, so that cross section goes along x-axis  
then x and y are n and t respectively scaled by width  
N and T coordinates

## 18.15 xy2sn

convert cartesian to streamwise coordiantes

## 18.16 xy2sn.java

use java port for speed up

## 18.17 xy2sn\_old

transform points from cartesian into streamwise coordinates

NOTE : prefer the java version, this has some problems with round off

# 19 linear-algebra

## 19.1 det2x2

2x2 matrix inverse of 2x2 matrices stacked along dim 3

## 19.2 det3x3

determinant of stacked 3x3 matrices

### 19.3 `det4x4`

determinant of stacked 4x4 matrices

### 19.4 `diag2x2`

diagonal of stacked 2x2 matrices

### 19.5 `eig2x2`

eigenvalues of stacked 2x2 matrices

## 20 linear-algebra/eigenvalue

### 20.1 `eig_bisection`

### 20.2 `eig_inverse`

### 20.3 `eig_inverse_iteration`

### 20.4 `eig_power_iteration`

## 21 linear-algebra/eigenvalue/jacobi-davidson

### 21.1 `afun_jdm`

## 21.2 davidson

## 21.3 jacobi\_davidson

## 21.4 jacobi\_davidson\_qr

## 21.5 jacobi\_davidson\_qz

## 21.6 jacobi\_davidson\_simple

## 21.7 jdqr

```
% Read/set parameters
% Initiate global variables
% Return if eigenvalueproblem is trivial
% Initialize V, W:
%   V,W orthonormal, A*V=W*R+Qschur*E, R upper triangular
% The JD loop (Standard)
%   V orthogonal, V orthogonal to Qschur
%   V*V=eye(j), Qschur'*V=0,
%   W=A*V, M=V'*W
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   Both V and W orthonormal and orthogonal w.r.t. Qschur
%    $V^*V = \text{eye}(j)$ ,  $Qschur' * V = 0$ ,  $W' * W = \text{eye}(j)$ ,  $Qschur' * W = 0$ 
%    $(A * V - \tau * V) = W * R + Qschur * E$ ,  $E = Qschur' * (A * V - \tau * V)$ ,  $M = W' * V$ 
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
  Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   V W AV.
%   Both V and W orthonormal and orthogonal w.r.t. Qschur,  $AV = A * V - \tau * V$ 
%    $V^*V = \text{eye}(j)$ ,  $W' * W = \text{eye}(j)$ ,  $Qschur' * V = 0$ ,  $Qschur' * W = 0$ ,
%    $(I - Qschur * Qschur') * AV = W * R$ ,  $M = W' * V$ ;  $R = W' * AV$ ;
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
  Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```



```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   W orthonormal, V and W orthogonal to Qschur,
%   W'*W=eye(j), Qschur'*V=0, Qschur'*W=0
%   W=(A*V-tau*V)-Qschur*E, E=Qschur'*(A*V-tau*V),
%   M=W'*V
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
  Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
  W=V*Q; V=V(:,1:j)/R; E=E/R; R=eye(j); M=Q(1:j,:)' /R;
  W=V*H; V(:,j+1)=[];R=R'*R; M=H(1:j,:)' ;
%===== ARNOLDI (for initializing spaces)
%=====
%===== END ARNOLDI
%=====

% not accurate enough M=Rw'\(M/Rv);
%===== COMPUTE SORTED JORDAN FORM
%=====

% compute vectors and matrices for skew projection
% solve preconditioned system
% 0 step of bicgstab eq. 1 step of bicgstab
% Then x is a multiple of b
% HIST=[0,1];
  explicit preconditioning
% compute norm in l-space
% HIST=[HIST;[nmv,rnm/snm]];
% sufficient accuracy. No need to update r,u
  implicit preconditioning
% collect the updates for x in l-space

```

```

% but, do the orth to Q implicitly
% compute norm in l-space
% HIST=[HIST;[nmv,rnrm/snrm]];
% sufficient accuracy. No need to update r,u
% Do the orth to Q explicitly
% In exact arithmetic not needed, but
% appears to be more stable.
% plot(HIST(:,1),log10(HIST(:,2)+eps),'*'), drawnow, pause
% 0 step of gmres eq. 1 step of gmres
% Then x is a multiple of b
%=====

% 0 step of gmres eq. 1 step of gmres
% Then x is a multiple of b
HIST=1;
% Lucky break-down
HIST=[HIST;(gamma~=0)/sqrt(rho)];
% Lucky break-down
% solve in least square sense
HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
plot(J,HIST(:,1),'*'); drawnow,% pause
r=r/rho; rho=1;
% HIST=rho;
% HIST=[HIST;rho];
HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
plot(J,HIST(:,1),'*'); drawnow,% pause
% HIST = rho;
% HIST=[HIST;rho];
HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
plot(J,HIST(:,1),'*'); drawnow, pause
% HIST = rho;
% HIST=[HIST;rho];
HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
plot(J,HIST(:,1),'*'); drawnow, pause
%----- compute schur form -----
A*Q=Q*S, Q'*Q=eye(size(A));
% transform real schur form to complex schur form
%----- find order eigenvalues -----
%----- reorder schur form -----
%----- compute qz form -----
%----- sort eigenvalues -----
%----- sort qz form -----
% i>j, move ith eigenvalue to position j
% determine dimension
% defaults
%% 'v'

```

## 21.8 jdqr\_sleijpen

```
% Read/set parameters
% Initiate global variables
% Return if eigenvalueproblem is trivial
% Initialize V, W:
%   V,W orthonormal,  $A*V=W*R+Qschur*E$ , R upper triangular
% The JD loop (Standard)
%   V orthogonal, V orthogonal to Qschur
%    $V*V=eye(j)$ ,  $Qschur'*V=0$ ,
%    $W=A*V$ ,  $M=W'*W$ 
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
  Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   Both V and W orthonormal and orthogonal w.r.t. Qschur
%    $V*V=eye(j)$ ,  $Qschur'*V=0$ ,  $W'*W=eye(j)$ ,  $Qschur'*W=0$ 
%    $(A*V-tau*V)=W*R+Qschur*E$ ,  $E=Qschur'*(A*V-tau*V)$ ,  $M=W'*V$ 
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
  Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   V W AV.
%   Both V and W orthonormal and orthogonal w.r.t. Qschur, AV=A*V-
%   tau*V
%   V*V=eye(j), W'*W=eye(j), Qschur'*V=0, Qschur'*W=0,
%   (I-Qschur*Qschur')*AV=W*R, M=W'*V; R=W'*AV;
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
%   Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   W orthonormal, V and W orthogonal to Qschur,
%   W'*W=eye(j), Qschur'*V=0, Qschur'*W=0
%   W=(A*V-tau*V)-Qschur*E, E=Qschur'*(A*V-tau*V),
%   M=W'*V
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
%   Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
W=V*Q; V=V(:,1:j)/R; E=E/R; R=eye(j); M=Q(1:j,:)' /R;
W=V*H; V(:,j+1)=[];R=R'*R; M=H(1:j,:)' ;
%===== ARNOLDI (for initializing spaces)
=====
%===== END ARNOLDI
=====
% not accurate enough M=Rw'\(M/Rv);
%===== COMPUTE SORTED JORDAN FORM
=====
% compute vectors and matrices for skew projection
% solve preconditioned system
% 0 step of bicgstab eq. 1 step of bicgstab
% Then x is a multiple of b
% HIST=[0,1];
% explicit preconditioning
% compute norm in l-space
% HIST=[HIST;[nmv,rnorm/snorm]];
% sufficient accuracy. No need to update r,u
% implicit preconditioning
% collect the updates for x in l-space
% but, do the orth to Q implicitly
% compute norm in l-space
% HIST=[HIST;[nmv,rnorm/snorm]];
% sufficient accuracy. No need to update r,u
% Do the orth to Q explicitly
% In exact arithmetic not needed, but
% appears to be more stable.
% plot(HIST(:,1),log10(HIST(:,2)+eps),'*'), drawnow, pause
% 0 step of gmres eq. 1 step of gmres
% Then x is a multiple of b
%=====

% 0 step of gmres eq. 1 step of gmres
% Then x is a multiple of b
HIST=1;
% Lucky break-down
HIST=[HIST;(gamma~=0)/sqrt(rho)];
% Lucky break-down
% solve in least square sense
HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
plot(J,HIST(:,1),'*'); drawnow,% pause
r=r/rho; rho=1;
% HIST=rho;
% HIST=[HIST;rho];
HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';

```

```

    plot(J,HIST(:,1),'*'); drawnow,% pause
% HIST = rho;
% HIST=[HIST;rho];
    HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
    plot(J,HIST(:,1),'*'); drawnow, pause
% HIST = rho;
% HIST=[HIST;rho];
    HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
    plot(J,HIST(:,1),'*'); drawnow, pause
%----- compute schur form -----
    A*Q=Q*S, Q'*Q=eye(size(A));
% transform real schur form to complex schur form
%----- find order eigenvalues -----
%----- reorder schur form -----
%----- compute qz form -----
%----- sort eigenvalues -----
%----- sort qz form -----
% i>j, move ith eigenvalue to position j
% determine dimension
% defaults
%% 'v'

```

## 21.9 jdqr\_vorst

```

% Read/set parameters
% Initiate global variables
% Return if eigenvalueproblem is trivial
% Initialize V, W:
%   V,W orthonormal, A*V=W*R+Qschur*E, R upper triangular
% The JD loop (Standard)
%   V orthogonal, V orthogonal to Qschur
%   V*V=eye(j), Qschur'*V=0,
%   W=A*V, M=V'*W
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
    Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   Both V and W orthonormal and orthogonal w.r.t. Qschur
%    $V^*V = \text{eye}(j)$ ,  $Qschur' * V = 0$ ,  $W' * W = \text{eye}(j)$ ,  $Qschur' * W = 0$ 
%    $(A * V - \tau * V) = W * R + Qschur * E$ ,  $E = Qschur' * (A * V - \tau * V)$ ,  $M = W' * V$ 
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
  Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   V W AV.
%   Both V and W orthonormal and orthogonal w.r.t. Qschur,  $AV = A * V - \tau * V$ 
%    $V^*V = \text{eye}(j)$ ,  $W' * W = \text{eye}(j)$ ,  $Qschur' * V = 0$ ,  $Qschur' * W = 0$ ,
%    $(I - Qschur * Qschur') * AV = W * R$ ,  $M = W' * V$ ;  $R = W' * AV$ ;
%
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
  Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
% The JD loop (Harmonic Ritz values)
%   W orthonormal, V and W orthogonal to Qschur,
%   W'*W=eye(j), Qschur'*V=0, Qschur'*W=0
%   W=(A*V-tau*V)-Qschur*E, E=Qschur'*(A*V-tau*V),
%   M=W'*V
% Compute approximate eigenpair and residual
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Check for convergence
% Expand the partial Schur form
Rschur=[Rschur;zeros(1,k)],Qschur'*MV(u)]; k=k+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Expand preconditioned Schur matrix PinvQ
% Check for shrinking the search subspace
% Solve correction equation
% Expand the subspaces of the interaction matrix
W=V*Q; V=V(:,1:j)/R; E=E/R; R=eye(j); M=Q(1:j,:)' /R;
W=V*H; V(:,j+1)=[];R=R'*R; M=H(1:j,:)' ;
%===== ARNOLDI (for initializing spaces)
=====
%===== END ARNOLDI
=====

% not accurate enough M=Rw'\(M/Rv);
%===== COMPUTE SORTED JORDAN FORM
=====

% accepted separation between eigenvalues:
% no preconditioning
% solve left preconditioned system
% compute vectors and matrices for skew projection
% precondition and project r
% solve preconditioned system
% no preconditioning
% solve two-sided expl. preconditioned system
% compute vectors and matrices for skew projection
% precondition and project r
% solve preconditioned system
% "unprecondition" solution
%%% u(:,j+1)=Atilde*u(:,j)

```



```

%%%% r(:,j+1)=Atilde*r(:,j)
%----- compute schur form -----
A*Q=Q*S, Q'*Q=eye(size(A));
% transform real schur form to complex schur form
%----- find order eigenvalues -----
%----- reorder schur form -----
%----- compute qz form -----
%----- sort eigenvalues -----
%----- sort qz form -----
% i>j, move ith eigenvalue to position j
% determine dimension
% defaults

```

## 21.10 jdqz

```

% Read/set parameters
% Return if eigenvalueproblem is trivial
% Initialize target, test space and interaction matrices
% V=RepGS(Qschur,V); [AV,BV]=MV(V); %%% more stability??
% W=RepGS(Zschur,eval(testspace)); %%% dangerous if sigma~lambda
% Solve the preconditioned correction equation
% Expand the subspaces and the interaction matrices
% Check for stagnation
% Solve projected eigenproblem
% Compute approximate eigenpair and residual
%=== an alternative, but less stable way of computing z =====
% display history
% save history
% check convergence
% EXPAND Schur form
% Expand preconditioned Schur matrix MinvZ=M\Zschur
% check for conjugate pair
% To detect whether another eigenpair is accurate enough
% restart if dim(V)> jmax
% Initialize target, test space and interaction matrices
% additional stabilisation. May not be needed
% V=RepGS(Zschur,V); [AV,BV]=MV(V);
% end add. stab.
% Solve the preconditioned correction equation
% expand the subspaces and the interaction matrices
% Check for stagnation
% compute approximate eigenpair
% Compute approximate eigenpair and residual
% display history
% save history
% check convergence
% expand Schur form

```

```

% ZastQ=Z'*Q0
% the final Qschur
% check for conjugate pair
% t perp Zschur, t in span(Q0,imag(q))
% To detect whether another eigenpair is accurate enough
% restart if dim(V)> jmax
%===== END JDQZ
=====
%=====
%===== PREPROCESSING
=====
%=====
%===== ARNOLDI (for initial spaces)
=====
%% then precondition=I and target = 0: apply Arnoldi with A
%===== END ARNOLDI
=====
%=====
%===== POSTPROCESSING
=====
%=====
%===== SORT QZ DECOMPOSITION INTERACTION MATRICES
=====
%===== COMPUTE SORTED JORDAN FORM
=====
%===== END JORDAN FORM
=====
%===== OUTPUT
=====
%=====
%===== UPDATE PRECONDITIONED SCHUR VECTORS
=====
%=====
%=====
%===== SOLVE CORRECTION EQUATION
=====
%=====
% solve preconditioned system
%=====

```

```

%===== LINEAR SOLVERS
=====

% [At,Bt]=MV(x); At=theta(2)*At-theta(1)*Bt;
% xtol=norm(r-At+Z*(Z'*At))/norm(r);
%===== Iterative methods
=====

% 0 step of bicgstab eq. 1 step of bicgstab
% Then x is a multiple of b
% HIST=[0,1];
% explicit preconditioning
% compute norm in l-space
% HIST=[HIST;[nmv,rnorm/snorm]];
% sufficient accuracy. No need to update r,u
% implicit preconditioning
% collect the updates for x in l-space
% but, do the orth to Z implicitly
% compute norm in l-space
% HIST=[HIST;[nmv,rnorm/snorm]];
% sufficient accuracy. No need to update r,u
% Do the orth to Z explicitly
% In exact arithmetic not needed, but
% appears to be more stable.
% plot(HIST(:,1),log10(HIST(:,2)+eps),'*'), drawnow
% 0 step of gmres eq. 1 step of gmres
% Then x is a multiple of b
%=====

% 0 step of gmres eq. 1 step of gmres
% Then x is a multiple of b
HIST=1;
% Lucky break-down
HIST=[HIST;(gamma~=0)/sqrt(rho)];
% Lucky break-down
% solve in least square sense
HIST=log10(HIST+eps); J=[0:size(HIST,1)-1]';
plot(J,HIST(:,1),'*'); drawnow
%===== END SOLVE CORRECTION EQUATION
=====

%===== BASIC OPERATIONS
=====

%=====

y(1:5,1), pause
%===== COMPUTE r AND z
=====

```

```

% E*u=Q*sigma, sigma(1,1)>sigma(2,2)
%===== END computation r and z
=====
%=====

%===== Orthogonalisation
=====
%=====

%===== END Orthogonalisation
=====
%=====

%===== Sorts Schur form
=====
%=====

        kappa=max(norm(A,inf)/max(norm(B,inf),1.e-12),1);
        kappa=2^(round(log2(kappa)));
%----- compute the qz factorization -----
%----- scale the eigenvalues -----
%----- sort the eigenvalues -----
%----- swap the qz form -----
% repeat SwapQZ if angle is too small
%=====

%=====

% i>j, move ith eigenvalue to position j
% compute q s.t. C*q=(t(i,1)*S-s(i,1)*T)*q=0
C*P=Q*R
check whether last but one diag. elt r nonzero
C*q
% end computation q
%===== END sort QZ decomposition interaction matrices
=====
%=====

%===== INITIALIZATION
=====
%=====

%=====

% defaults          %%%% search for 'xx' in fieldnames
%% 'ma'
%% 'sch'
%% 'to'
%% 'di'

```

```

% jmin=nselect+p0 %%%% 'jmi'
% jmax=jmin+p1 %%%% 'jma'
%% 'te'
%% 'pai'
%% 'av'
%% 'tr'
%% 'fix'
%% 'ns'
%% 'ch'
%% 'lso'
%% 'ls_m'
%% 'ls_t'
%% 'ls_e'
%% 'ty'
%% 'l_'
%% 'u_'
%% 'p_'
%% 'sca'
%% 'v0'
initiation
'standard'
'harmonic'
'searchspace'
%=====

% or Operator_Form=3 or Operator_Form=5???
%=====

%===== DISPLAY FUNCTIONS
=====
%=====

%=====

%=====

%=====

```

## 21.11 mfunc\_jdm

## 21.12 mgs

### 21.13 minres\_

### 21.14 mv\_jacobi\_davidson

## 22 linear-algebra

### 22.1 first

### 22.2 gershgorin\_circle

range of eigenvalues determined by the gershgorin circle theorem

### 22.3 haussdorff

haussdorf dimension

box counting: count rectangles passed through by line (covered by polygon)

Koch snow flake 3:4 -> 1.2619

Kantor set 2:3, (4:9) -> 0.6309

quadrat 4:2, 9:3, 16:4 -> 2

### 22.4 eig2x2

reconstruct matrix from eigenvalue decomposition

### 22.5 inv2x2

2x2 inverse of stacked matrices

## 22.6 inv3x3

## 22.7 inv4x4

inverse of stacked 4x4 matrices

# 23 linear-algebra/lanczos

## 23.1 arnoldi

## 23.2 arnoldi\_new

## 23.3 eigs\_lanczos\_man

## 23.4 lanczos

## 23.5 lanczos\_

## 23.6 lanczos\_biorthogonal

## 23.7 lanczos\_biorthogonal\_improved

**23.8**   `lanczos_ghep`

**23.9**   `mv_lanczos`

**23.10**   `reorthogonalise`

**23.11**   `test_lanczos`

## **24**   `linear-algebra/linear-systems`

**24.1**   `gmres_man`

`break on convergence`

**24.2**   `minres_recycle`

## **25**   `linear-algebra`

**25.1**   `lpmean`

`mean of pth-power of a`

**25.2**   `lpnorm`

`norm of lth-power of a`



### 25.3 matvec3

matrix-vector product of stacked matrices and vectors

### 25.4 max2d

maximum value and i-j index for matrix

### 25.5 mid

mid point between neighbouring vector elements

### 25.6 mpoweri

approximation of  $A^p$ , where  $p$  is not integer by quadratic interpolation

### 25.7 mtimes2x2

### 25.8 mtimes3x3

product of stacked 3x3 matrices

### 25.9 nannorm

norm of a vector, skips nan-values

### 25.10 nanshift

shift vector, but set out of range values to NaN

### 25.11 nl

number rows (lines) of a matrix

analogue to unix nl command

### 25.12 normalise

normalise a vector or the columns of a matrix

note that the columns are independently normalised, and hence not necessarily

orthogonal to each other use the gram schmidt algorithm for this (qr or orth)

### 25.13 normalize1

normalize columns in x to [-1,1]

### 25.14 normrows

### 25.15 orth2

make matrix A orthogonal to B

### 25.16 orth\_man

orthogonalize the columns of A

### 25.17 orthogonalise

make x orthogonal to Y

## 25.18 paddext

padd values to vactor  
not suitable for noisy data  
order = 0 : constant extrapolation (hold)  
order = 1 : linear extrapolation

## 25.19 paddval1

padd values at end of x

## 25.20 paddval2

padd values to x

# 26 linear-algebra/polynomial

## 26.1 chebychev

chebycheff polynomials

## 26.2 piecewise\_polynomial

evaluate piecewise polynomial

## 26.3 roots1

roots of linear functions

## 26.4 roots2

roots of quadratic function  
 $c_1 x^2 + c_2 x + c_3 = 0$

**26.5** roots2poly

**26.6** roots3

**26.7** roots4

**26.8** roots\_piecewise\_linear

**26.9** test\_roots4

**26.10** vanderi\_1d

vandermonde matrix of an integral

**27** linear-algebra

**27.1** randrot

random rotation matrix

**27.2** right

get right column by shifting columns to left  
extrapolate rightmost column

### 27.3 rot2

rotation matrix from angle

### 27.4 rot2dir

rotation matrix from direction vector

### 27.5 rot3

### 27.6 rotR

### 27.7 rownorm

### 27.8 simmilarity\_matrix

### 27.9 spnorm

frobenius norm

### 27.10 spzeros

allocate a sparze matrix of zeros

### 27.11 test\_roots3

## 27.12 transform\_minmax

## 27.13 transpose3

transpose stacked 3x3 matrices

## 27.14 transposeall

# 28 logic

bitwise operations on integers

## 28.1 bitor\_man

bitwise OR of the numbers of the columns of A

input:  
A (positive integer)

# 29 master/plot

## 29.1 attach\_boundary\_value

## 29.2 cartesian\_polar

## 29.3 img\_vargrid

29.4 `plot_basis_functions`

29.5 `plot_convergence`

29.6 `plot_dof`

29.7 `plot_eigenbar`

29.8 `plot_error_estimation`

29.9 `plot_error_estimation_2`

29.10 `plot_error_fem`

29.11 `plot_fdm_kernel`

29.12 `plot_fdm_vs_fem`

29.13 `plot_fem_accuracy`

29.14 `plot_function_and_grid`

29.15 `plot_hat`

29.16 `plot_hydrogen_wf`

29.17 `plot_mesh`

29.18 `plot_mesh_2`

29.19 `plot_refine`

29.20 `plot_refine_3d`

29.21 `plot_runtime`

29.22 `plot_spectrum`

29.23 `plot_wavefunction`



## 30 master/portcd

### 30.1 assemble\_2d\_phi\_phi

### 30.2 assemble\_3d\_dphi\_dphi

### 30.3 assemble\_3d\_phi\_phi

### 30.4 dV\_2d\_

### 30.5 derivative\_2d

### 30.6 derivative\_3d

### 30.7 element\_neighbour\_2d

### 30.8 prefetch\_2d\_

### 30.9 promote\_2d\_3\_10

30.10 promote\_2d\_3\_15

30.11 promote\_2d\_3\_21

30.12 promote\_2d\_3\_6

30.13 promote\_3d\_4\_10

30.14 promote\_3d\_4\_20

30.15 promote\_3d\_4\_35

30.16 vander\_2d

30.17 vander\_3d

## 31 number-theory

31.1 ceiln

floor to leading n-digits

## 31.2 digitsb

number of digits with respect to specified base

## 31.3 floorn

floor to n-digits

## 31.4 iseven

true for even numbers in X

## 31.5 multichoosek

all combinations of length k from set values with repetitions  
c.f. nchoosek, combinations without repetition

input :  
    x : scalar integer or vector of arbitrary numbers  
    k : length of subsets  
output :  
    if x scalar : number of combinations  
    if x vector : the exact combinations

## 31.6 nchoosek\_man

vectorised binomial coefficient  
 $b = N! / K! (N-K)!$

## 31.7 pythagorean\_triple

pythagorean triple

## 31.8 roundn

round to n digits

## **32 numerical-methods/differentiation**

### **32.1 derivative1**

first derivative on variable mesh  
second order accurate

### **32.2 derivative2**

second derivative on a variable mesh

## **33 numerical-methods/finite-difference**

### **33.1 cdiff**

differences of columns of X  
degree = 1 : central first order differences  
degree = 2 : central second order differences

### **33.2 cdiffb**

differences of columns of X  
degree = 1 : central first order differences  
degree = 2 : central second order differences  
TODO use difference matrix function for simplicity

### **33.3 central\_difference**

### **33.4 cmean**

single gaussian smoothing step with kernel  $1/4*[1,2,1]$

### **33.5 cmean2**

### 33.6 derivative\_matrix\_1\_1d

finite difference matrix of first derivative in one dimensions  
n : number of grid points  
h = L/(n+1) constant step with  
function [D1, d1] = derivative\_matrix\_1d(n,L,order)

### 33.7 derivative\_matrix\_2\_1d

finite derivative matrix of second derivative in one dimension

### 33.8 derivative\_matrix\_2d

finite difference derivative matrix in two dimensions

### 33.9 derivative\_matrix\_curvilinear

derivative matrix on a curvilinear grid

### 33.10 derivative\_matrix\_curvilinear\_2

derivative matrix on a two dimensional curvilinear grid  
the grid has not necessarily to be orthogonal

### 33.11 difference\_kernel

difference kernels for equispaced grids  
c.f. Computing the Spectrum of the Confined Hydrogen Atom, Kastner,  
2012

### 33.12 distmat

distance matrix for a 2 dimensional rectangular matrix

### 33.13 downwind\_difference

### 33.14 gradpde2d

objective function gradient on two dimensional regular grid  
numeric gradient for non-linear least squares optimisation  
of a PDE on a rectangular grid  
 $x_* = \min(f(x))$   
 $f = (v(x) - v(x_*))^2 = f(x) + A \, dx + O(dx^2)$   
 $a_{ij} = df_i/dx_j$

### 33.15 laplacian

### 33.16 laplacian\_fdm

finite difference matrix of the laplacian  
BC

### 33.17 left

left element of vector, leftmost column is extrapolated

### 33.18 lrmean

mean of the left and right element

## 34 numerical-methods/finite-difference/master

### 34.1 fdm\_adaptive\_grid

34.2 `fdm_adaptive_refinement_old`

34.3 `fdm_assemble_d1_2d`

34.4 `fdm_assemble_d2_2d`

34.5 `fdm_confinement`

34.6 `fdm_d_vargrid`

34.7 `fdm_h_unstructured`

34.8 `fdm_hydrogen_vargrid`

34.9 `fdm_mark_unstructured_2d`

34.10 `fdm_plot`

34.11 `fdm_plot_series`

**34.12** `fdm_refine_2d`

**34.13** `fdm_refine_3d`

**34.14** `fdm_refine_unstructured_2d`

**34.15** `fdm_schroedinger_2d`

**34.16** `fdm_schroedinger_3d`

**34.17** `relocate`

## **35 numerical-methods/finite-difference**

**35.1** `mid`

`mid` point between neighbouring vector elements

**35.2** `pwmid`

segment end point to segment mid point transformation for regular 1  
d grids



### **35.3 ratio**

ratio of two subsequent values

### **35.4 steplength**

step length of a vector if it were equispaced

### **35.5 swapoddeven**

swap odd and even elements in a vector

### **35.6 test\_derivative\_matrix\_2d**

### **35.7 test\_derivative\_matrix\_curvilinear**

### **35.8 test\_difference\_kernel**

### **35.9 upwind\_difference**

## **36 numerical-methods/finite-element**

### **36.1 Mesh\_2d.java**

### **36.2 Tree\_2d.java**

36.3 assemble\_1d\_dphi\_dphi

36.4 assemble\_1d\_phi\_phi

36.5 assemble\_2d\_dphi\_dphi\_java

36.6 assemble\_2d\_phi\_phi\_java

36.7 assemble\_3d\_dphi\_dphi\_java

36.8 assemble\_3d\_phi\_phi\_java

36.9 boundary\_1d

36.10 boundary\_2d

36.11 boundary\_3d

36.12 check\_area\_2d

**36.13**   **circmesh**

**36.14**   **cropradius**

**36.15**   **display\_2d**

**36.16**   **display\_3d**

**36.17**   **distort**

**36.18**   **err\_2d**

**36.19**   **estimate\_err\_2d\_3**

**36.20**   **example\_1d**

**36.21**   **example\_2d**

**36.22**   **explode**

**36.23**   `fem_2d`

**36.24**   `fem_2d_heuristic_mesh`

**36.25**   `fem_get_2d_radial`

**36.26**   `fem_interpolation`

**36.27**   `fem_plot_1d`

**36.28**   `fem_plot_1d_series`

**36.29**   `fem_plot_2d`

**36.30**   `fem_plot_2d_series`

**36.31**   `fem_plot_3d`

**36.32**   `fem_plot_3d_series`

**36.33**   `fem_plot_confine_series`

**36.34**   `fem_radial`

`adaptive grid`  
`constant grid`

**36.35**   `flip_2d`

**36.36**   `get_mesh_arrays`

**36.37**   `hashkey`

**37**   `numerical-methods/finite-element/int`

**37.1**   `int_1d_gauss`

**37.2**   `int_1d_gauss_1`

**37.3**   `int_1d_gauss_2`

**37.4**   `int_1d_gauss_3`

37.5 int\_1d\_gauss\_4

37.6 int\_1d\_gauss\_5

37.7 int\_1d\_gauss\_6

37.8 int\_1d\_gauss\_lobatto

37.9 int\_1d\_gauss\_n

37.10 int\_1d\_nc\_2

37.11 int\_1d\_nc\_3

37.12 int\_1d\_nc\_4

37.13 int\_1d\_nc\_5

37.14 int\_1d\_nc\_6

37.15 int\_1d\_nc\_7

37.16 int\_1d\_nc\_7\_hardy

37.17 int\_2d\_gauss\_1

37.18 int\_2d\_gauss\_12

37.19 int\_2d\_gauss\_13

37.20 int\_2d\_gauss\_16

37.21 int\_2d\_gauss\_19

37.22 int\_2d\_gauss\_25

37.23 int\_2d\_gauss\_3

37.24 int\_2d\_gauss\_33

37.25 int\_2d\_gauss\_4

37.26 int\_2d\_gauss\_6

37.27 int\_2d\_gauss\_7

37.28 int\_2d\_gauss\_9

37.29 int\_2d\_nc\_10

37.30 int\_2d\_nc\_15

37.31 int\_2d\_nc\_21

37.32 int\_2d\_nc\_3

37.33 int\_2d\_nc\_6

37.34 int\_3d\_gauss\_1



37.35 int\_3d\_gauss\_11

37.36 int\_3d\_gauss\_14

37.37 int\_3d\_gauss\_15

37.38 int\_3d\_gauss\_24

37.39 int\_3d\_gauss\_4

37.40 int\_3d\_gauss\_45

37.41 int\_3d\_gauss\_5

37.42 int\_3d\_nc\_11

37.43 int\_3d\_nc\_4

37.44 int\_3d\_nc\_6

37.45 int\_3d\_nc\_8

## 38 numerical-methods/finite-element

38.1 interpolation\_matrix

38.2 mark

38.3 mark\_1d

38.4 mesh\_1d\_uniform

38.5 mesh\_3d\_uniform

38.6 mesh\_interpolate

38.7 neighbour\_1d

38.8 old

38.9 pdeeig\_1d

38.10 pdeeig\_2d

38.11 pdeeig\_3d

38.12 polynomial\_derivative\_1d

38.13 potential\_const

38.14 potential\_coulomb

38.15 potential\_harmonic\_oscillator

38.16 project\_circle

38.17 project\_rectangle

38.18 promote\_1d\_2\_3

38.19 promote\_1d\_2\_4

38.20 promote\_1d\_2\_5

38.21 promote\_1d\_2\_6

38.22 quadrilaterate

38.23 recalculate\_regularity\_2d

38.24 refine\_1d

38.25 refine\_2d\_21

38.26 refine\_2d\_structural

38.27 regularity\_1d

38.28 regularity\_2d

### 38.29 regularity\_3d

```
{      T = [1 2 3 4];  
}
```

### 38.30 relocate\_2d

### 38.31 test\_circmesh

### 38.32 test\_hermite

### 38.33 tri\_assign\_points

### 38.34 triangulation\_uniform

### 38.35 vander\_1d

van der Monde matrix

### 38.36 vanderd\_1d

### 38.37 vanderi\_1d

## 39 numerical-methods/finite-volume/@Advection

### 39.1 Advection

FVM treatment of the Advection equation

### 39.2 dot\_advection

advection equation

## 40 numerical-methods/finite-volume/@Burgers

### 40.1 burgers\_split

viscous Burgers' equation,  
mixed analytic and numerical derivative in frequency space  
by splitting scheme  
 $u_t = -(0.5*u^2)_x + c*u_{xx}$

### 40.2 dot\_burgers\_fdm

viscous burgers' equation  
 $u_t = -d/dx (1/2*u^2) + c d^2/dx^2 u_{xx}$

### 40.3 dot\_burgers\_fft

viscous Burgers' equation in frequency space  
 $u_t + (0.5*u^2)_x = c*u_{xx}$

## 41 numerical-methods/finite-volume/@Finite\_Volume

### 41.1 Finite\_Volume

finite volume method for partial differential equations 1+1  
dimensions  
(time and space)

## 41.2 apply\_bc

apply boundary conditions

## 41.3 solve

solve the the PDE by successively stepping in time  
this is a trivial implmentation with constant step length  
severity of diffusive error depends on dt/dx-ratio  
stability depends on wave height

```
printf('Progress %2.1f%% %2.1fs\n',100*(t-Ti
(1))/(Ti(2)-Ti(1)),t_real);
```

## 41.4 step\_split\_strang

step in time, treat inhomogeneous part by Strang splitting  
this scheme is not suitable for stationary solutions, for example  
steady shallow water flow

## 41.5 step\_unsplit

step in time, without splitting the inhomogeneous term

# 42 numerical-methods/finite-volume/@Flux\_Limiter

## 42.1 Flux\_Limiter

class of flux limiters

## 42.2 beam\_warming

beam warming scheme  
low resolution  
note: works only if sign of eigenvalues point into the same  
direction according to RL

### 42.3 fromm

fromme limiter  
low res

### 42.4 lax\_wendroff

lax wendroff scheme  
second order accurate, but no tvd  
this is effectively not a limiter  
eq. 6.39 in randall, leveque

### 42.5 minmod

min-mod schock limiter

### 42.6 monotized\_central

monotonized central flux limiter

### 42.7 muscl

muscl flux limiter

### 42.8 superbee

superbee limiter

### 42.9 upwind

godunov scheme  
godunov, first order accurate



## 42.10 vanLeer

van Leer limiter

## 43 numerical-methods/finite-volume/@KDV

### 43.1 dot\_kdv\_fdm

korteweg de vries equation  
 $u_t + (0.5*u^2)_x = c*u_{xxx}$

### 43.2 dot\_kdv\_fft

korteweg de vries equation  
compute derivatives in frequency space  
 $u_t + (0.5*u^2)_x = c*u_{xxx}$

### 43.3 kdv\_split

korteweg de vries equation in frequency space,  
derivative treated by splitting scheme

## 44 numerical-methods/finite-volume/@Reconstruct\_Average\_Evolve

### 44.1 Reconstruct\_Average\_Evolve

Reconstruct Average Evolve Finite Volume Method for treatment of  
1+1D pdes

McCronack Scheme

err =  $O(dt^2) + O(dx^2)$ , except as discontinuities  
error:

```
h_xxx(3:end-2) = 1/dx^3*( -0.5*h(1:end-4) + h(2:end-3) - h(4:
    end-1) + 0.5*h(5:end) );
th = -1/6*dx^2*qh_.*(1 - (qh_*dt/dx).^2).*h_xxx;
```

## 44.2 advect\_highres

single time step for the reconstruct evolve algorithm

## 44.3 advect\_lowres

single time step  
low resolution

# 45 numerical-methods/finite-volume

## 45.1 Godunov

Godunov, upwind method for systems of pdes

## 45.2 Lax\_Friedrich

Lax-Friedrich-Method  
for hyperbolic conservation laws  
 $\text{err} = O(\text{dt}) + O(\text{dx})$   
 $|a \text{ dt/dx}| < 1$

## 45.3 Measure

## 45.4 Roe

non linear roe solver for the SWE (randall, leveque 15.3.1)

The roe solver guarantess:

- A is diagonalisable with real eigenvalues (15.12)
- can be determined by a closed formula
- is an efficient replacement for true Rieman solver

## 45.5 fv\_swe

wrapper for solving SWE

## 45.6 staggered\_euler

forward euler method with staggered grid

## 45.7 staggered\_grid

staggered grid approximation to the SWE

# 46 numerical-methods

## 46.1 grid2quad

extract rectangular elements of a structured grid  
in form of an unstructured quad-mesh format

# 47 numerical-methods/integration

## 47.1 cumintL

cumulative integral from left to right

## 47.2 cumintR

cumulative integral from right to left

## 47.3 int\_trapezoidal

integrate y along x with the trapezoidal rule

## 48 numerical-methods/interpolation/@Kriging

### 48.1 Kriging

```
class for Kriging interpolation
```

### 48.2 estimate\_semivariance

```
estimate the parameter of the semivariance model for Kriging
interpolation
    % set up the regression matrix and solve for
    parameters
```

### 48.3 interpolate\_

```
interpolate with Krieking method

this function may interpolate several quantities per coordinate,
using the same variogram, if the semivariance of the quantities
differs,
the user may prefer to estimate the semivariance and interpolate
each quantity
individually

Xs : source point coordinates
Vs : value at source points
Xt : target point coordinates
Vt : value at target points
E2t : squared interpolation error at target points
```

## 49 numerical-methods/interpolation/@RegularizedInterpolator

### 49.1 RegularizedInterpolator1

```
class for regularized interpolation (Thikonov) on a 1D mesh
```

### 49.2 init

```
initialize the interpolator with a set of sampling points
```

## **50 numerical-methods/interpolation/@RegularizedInterpolator2**

### **50.1 RegularizedInterpolator2**

```
class for regularized interpolation on an unstructures mesh (  
    interpolation)
```

#### **50.2 init**

```
initialize the interpolator with a set of point samples
```

## **51 numerical-methods/interpolation/@RegularizedInterpolator3**

### **51.1 RegularizedInterpolator3**

```
class for regularized interpolation (Tikhonov) on a triangulation  
(unstructured mesh)
```

#### **51.2 init**

```
initialize the interpolator with a set of sampling points
```

## **52 numerical-methods/interpolation**

### **52.1 IDW**

```
spatial averaging by inverse distance weighting
```

### **52.2 IPoly**

```
polynomial interpolation class
```

### 52.3 IRBM

```
interpolate by the radial basis function method
    fprintf(1,'Progress IRBM: %d%%\n',round(100*
        idx/size(Xi,1)));
```

### 52.4 ISparse

```
sparse interpolation class
```

### 52.5 Inn

```
nearest neighbour interpolation
```

### 52.6 Interpolator

```
interpolator super-class
    fprintf(1,'Progress: %f%% %fs\n',100*
        idx/size(Xt,1),t);
```

### 52.7 fixnan

```
fill nan-values in vector with gaps
```

### 52.8 idw1

```
spatial average by inverse distance weighting
```

### 52.9 idw2

```
spatial average by inverse distance weighting
```

## 52.10 inner2outer

linear interpolation of segment mid point to grid points at segment ends  
assumes equal grid spacing

## 52.11 inner2outer2

interpolate from element (segment) centres to edge points

## 52.12 interp1\_limited

interpolate values, but not beyond a certain distance  
this function is idempotent, i.e. it will not extrapolate over into gaps  
exceedint the limit and thus not spuriously extend the series when called a second time on the same data

## 52.13 interp1\_man

interpolate

## 52.14 interp1\_piecewise\_linear

## 52.15 interp1\_save

make interpolation save to round off errors  
the matlab internal interpolation suffers from rounding errors, which are unacceptable when values of X and Y are large (for example UTM coordinates)  
this normalization prevents this

## 52.16 interp1\_slope

quadratic interpolation returning value and derivative(s)

## 52.17 interp1\_smooth

## 52.18 interp1\_unique

matlab fails to interpolate, when x values are not unique  
this function makes the values unique before use

## 52.19 interp2\_man

nearest neighbour interpolation in two dimensions

## 52.20 interp\_angle

interpolate an angle

## 52.21 interp\_fourier

interpolation by the fourier method

## 52.22 interp\_fourier\_batch

batch interpolation by the fourier interpolation



### 52.23 interp\_sn

interpolate along streamwise coordinates  
This gives similar result to setting aspect ratio for sN to  
infinity,  
but not quite, as the input point set is not dense (scale for sN to  
infinity does not work)  
    sdx = sdx(sdx\_);

### 52.24 interp\_sn2

interpolation in streamwise coordinates

### 52.25 interp\_sn3

### 52.26 interp\_sn\_

### 52.27 limit\_by\_distance\_1d

smooth subsequent values along a curve such that  
     $v(x_0+dx) < v(x_0) + (ratio-1)*dx$   
if  $v$  is the edge length in a resampled polygon, then  $v_i/v_{(i+1)} <$   
    ratio  
     $ratio^{-1} = \exp(a*1)$

### 52.28 resample1

interpolation along a parametric curve with variable step width

### 52.29 resample\_d\_min

resample a function

### 52.30 resample\_vector

resample a track so that velocity vectors do not run into each other

### 52.31 test\_interp1\_limited

## 53 numerical-methods

### 53.1 inverse\_complex

### 53.2 maccormack\_step

## 54 numerical-methods/ode/@BVPS\_Characteristic

### 54.1 BVPS\_Characteristic

### 54.2 assemble1\_A

### 54.3 assemble1\_A\_Q

### 54.4 assemble2\_A

54.5 assemble\_AA

54.6 assemble\_AAA

54.7 bvp1c

54.8 check\_arguments

54.9 couple\_junctions

54.10 derivative

54.11 init

54.12 reconstruct

54.13 resample

## 54.14 solve

solve system of non-linear second order odes (in more than one variable)  
as boundary value problems

odefun provides ode coefficients c:  
$$c(x,1) y''(x) + c(x,2) y'(x) + c(x,3) y = c(x,4)$$
$$c_1 y'' + c_2 y' + c_3 y + c_4 = c_4$$

subject to the boundary conditions  
bcfun provides v and p and optionally q, so that:

$$b_1 y + b_2 y' = f$$
$$q(x,1)*(p(x,1) y_l(x) + p(x,2) y_l'(x) + q(x,2)*(p(x,1) y_r(x) + p(x,2) y_r'(x) = v(x)$$

where q weighs the waves travelling from left to right and right to left (default [1 1])

## 55 numerical-methods/ode/@Time\_Stepper

### 55.1 Time\_Stepper

### 55.2 solve

## 56 numerical-methods/ode

### 56.1 bvp2fdm

solve system of non-linear second order odes (in more than one variable)  
as boundary value problems by the finite difference method

odefun provides ode coefficients c:  
$$c(x,1) y''(x) + c(x,2) y'(x) + c(x,3) y = c(x,4)$$
$$c_1 y'' + c_2 y' + c_3 y + c_4 = 0$$

subject to the boundary conditions  
bcfun provides v and p and optionally q, so that:

```

b_1 y + b_2 y' = f
q(x,1)*( p(x,1) y_l(x) + p(x,2) y_l'(x)
+ q(x,2)*( p(x,1) y_r(x) + p(x,2) y_r'(x) = v(x)
where q weighs the waves travelling from left to right and right to
left (default [1 1])

```

## 56.2 bvp2wavetrain

solve second order boundary value problem by repeated integration

## 56.3 bvp2wavetwopass

two pass solution for the linearised wave equation  
solve first for the wave number k, and then for y

## 56.4 ivp\_euler\_forward

solve intial value problem by the euler forward method

## 56.5 ivprk2

solve initial value problem by the two step runge kutta method

## 56.6 ode2\_matrix

transformation matrix of second order ode  
to left and right going wave

```

c = odefun(x)
c1 y'' + c2' y + c3 y == 0
y = y_p + y_m, left and right going wave
d/dx [y_p, y_m] = A*[y_m, y_p]

```

## 56.7 ode2characteristic

second order odes  
transmittded and reflected wave

## **56.8 step\_trapezoidal**

single trapezoidal step

## **56.9 test\_bvp2**

# **57 numerical-methods/optimisation**

## **57.1 armijo\_stopping\_criterion**

armijo stopping criterion for optimizations

## **57.2 astar**

astar path finding algorithm

## **57.3 binsearch**

binary search on a line

## **57.4 bisection**

bisection

## **57.5 box1**

test objective function for optimisation routines

## **57.6 box2**

## 57.7 cauchy

## 57.8 cauchy2

solve non-linear system by cuachy's method  
slower than quadratic optimisation, but does not require a hessian  
fun : objective function, returns  
    f : scalar, objective function value  
    g : nx1, gradient  
x : nx1, initial position  
opt : options

## 57.9 directional\_derivative

directional (projected) derivative  
d : derivative, highest first  
p : series expansion around x0

## 57.10 dud

optimization by the dud algorithm

## 57.11 extreme3

extract maxima by quadratic approximation from sampled function val  
(t)  
intended to be called after [mval, mid] = max(val) for refinement  
of  
location and maximum

input  
t : sampling time (uniformly spaced)  
v : values at sampling times  
ouput:  
tdx : index where extremum should be computed  
t0 : location of the extremum  
val0 : value of extremum

$v'(dt0) = 0$  and  $v''(dt0)$  determines type of extremum

**57.12**   `extreme_quadratic`

**57.13**   `ftest`

**57.14**   `fzero_bisect`

**57.15**   `fzero_newton`

**57.16**   `grad`

`numerical gradient`

**57.17**   `hessian`

`numerical hessian`

**57.18**   `hessian_from_gradient`

`numerical hessian from gradient`

**57.19**   `hessian_projected`

`numerical hessian projected to one dimension`

**57.20**   `line_search`

`bisection routine`



### 57.21 line\_search2

bisection method

fun : objective funct  
x0 : start value  
f0 : objective function value at x0  
g : gradient at x0  
p : search direction from x0 (p = g for steepest descend)  
h : initial step length (default 1)  
lb : lower bound for x  
up : upper bound for x

### 57.22 line\_search\_polynomial

polynomial line search  
fun : objective funct  
x0 : start value  
f0 : objective function value at x0  
g : gradient at x0  
dir : search direction from x0 (p = g for steepest descend)  
h : initial step length (default 1)  
lb : lower bound for x  
up : upper bound for x

### 57.23 line\_search\_polynomial2

cubic line search  
fun : objective funct  
x0 : start value  
f0 : objective function value at x0  
g : gradient at x0  
dir : search direction from x0 (p = g for steepest descend)  
h : initial step length (default 1)  
lb : lower bound for x  
up : upper bound for x

### 57.24 line\_search\_quadratic

quadratic line search  
fun : objective funct  
x0 : start value

f0 : objective function value at x0  
g : gradient at x0  
dir : search direction from x0 (p = g for steepest descend)  
h : initial step length (default 1)  
lb : lower bound for x  
up : upper bound for x

### 57.25 line\_search\_quadratic2

quadratic line search

### 57.26 line\_search\_wolfe

line search by wolfe method  
c.f.: OPTIMIZATION THEORY AND METHODS - Nonlinear Programming, Sun,  
Yuan

### 57.27 ls\_bgfs

least squares by the bgfs method

### 57.28 ls\_broyden

least squares by the broyden method  
for rectangular / non symmetric systems  
Numerical Optimization nokedal  
Practical Methods of Optimization fletcher  
c.f. gerber 1981  
c.f. fletcher 1978 (more advanced, not used here)  
c.f. Kelley 1999 ch. 4

BGFS:  
Broyden 1965  
Fletcher 1970  
Goldfarb 1970  
Shanno 1970

### 57.29 ls\_generalized\_secant

least squares by the secant method  
Barnes, 1965  
Wolfe, 1959  
Fletcher 1980, 6.3  
seber 2003  
gerber

### 57.30 nlcg

non-linear conjugate gradient  
input:  
x : nx1 start vectort  
opt : struct options  
fdx : gradient constraint

### 57.31 nlls

non-linear least squares

### 57.32 picard

picard iteration

### 57.33 poly\_extrema

extrema of a polynomial

### 57.34 quadratic\_function

evaluate quadratic function in higher dimensions

### 57.35 quadratic\_programming

optimize by quadratic programming

### 57.36 quadratic\_step

single step of the quadratic programming

### 57.37 rosenbrock

rosenbrock test function

### 57.38 sqrt\_heron

Heron's method for the square root

### 57.39 test\_directional\_derivative

### 57.40 test\_dud

### 57.41 test\_fzero\_newton

### 57.42 test\_line\_search\_quadratic2

### 57.43 test\_ls\_generalized\_secant

### 57.44 test\_nlcg\_6\_order

## 57.45 test\_nlls

```
f = w'*(p*abs(x-1).^4) + w'*(1-p)*abs(x-1).^2;
```

## 58 numerical-methods/pde

### 58.1 laplacian2d\_fundamental\_solution

## 59 numerical-methods/piecewise-polynomials

### 59.1 Hermite1

hermite polynomial interpolation in 1d

### 59.2 hp2\_fit

fit a hermite polynomial  
coefficients are derivative free  
x0 : left point of first segment  
x1 : right point of last segment  
n : number of segments  
x : sample x-value  
val : sample y-value  
c : coefficients (values at points, no derivatives)

### 59.3 hp2\_predict

prediction with pw hermite polynomial  
c are values at support points

### 59.4 hp\_predict

predict with piecewise hermite polynomial

## 59.5 hp\_regress

fit piecewise hermite polynomial  
coefficients are values and derivatives

## 59.6 lp\_count

lagrangian basis for interpolation  
count number of valid samples

## 59.7 lp\_predict

lagrangian basis piecwie interpolation, predicor

## 59.8 lp\_regress

## 59.9 lp\_regress\_

# 60 numerical-methods

## 60.1 test\_adams\_bashforth

# 61 regression/@PolyOLS

## 61.1 PolyOLS

class for polynomial least squares

## 61.2 coefftest

### 61.3 detrend

detrending by polynomial regression

### 61.4 fit

fit a polynomial function  
like polyfit, but returns parameter error estimates  
TODO automatically activate scaleflag

### 61.5 fit\_

fit a polynomial function

### 61.6 predict

predict polynomial function values

### 61.7 predict\_

### 61.8 slope

slope by linear regression

## 62 regression/@PowerLS

### 62.1 PowerLS

class for power law regression

## 62.2 fit

fit a power law  
like polyfit, but returns parameter error estimates

## 62.3 predict

predict with power law  
S2 = diag((A\*obj.C)\*A');  
L = Y - S;  
U = Y + S;

## 62.4 predict\_

# 63 regression/@Theil

## 63.1 Theil

Kendal-Theil-Sen robust regression

## 63.2 detrend

linear detrending of a set of samples by the Theil-Senn Slope

## 63.3 fit

fit slope and intercept to a set of sample with the Theil-Sen  
method

c : confidence interval  $c = 2*ns*normcdf(1)$  for ns-sigma  
intervals  
param : itercept and slope  
P : confidence interval



## 63.4 predict

predict values and confidence intervals with the Theil-Sen method

## 63.5 slope

fit the slope with the Theil-Sen method

# 64 regression

linear and non-linear regression

## 64.1 Theil\_Multivariate

extension of the Theil-Senn regression to higher dimensions by means of the Gauss-Seidel iteration

## 64.2 areg

regression using the pth-fraction of samples with smallest residual

## 64.3 ginireg

gini regression

## 64.4 hesssimplereg

hessian, gradient and objective function value of the simple  
regression  
 $\text{rhs} = p(1) + p(2) x + \text{eps}$

## 64.5 l1lin

solve  $\|Ax - b\|_{L1}$  by means of linear programming

## 64.6 lsq\_sparam

parameter covariance of the least squares regression

fun : model function for prediction  
b : sample values  
 $f(p) = b$   
p : parameter at point of evaluation (preferably optimum)

## 64.7 polyfitd

fit a polynomial of order n to a set of sampled values and sampled values of the derivative

x0 must contain at least for conditioning as otherwise the intercept cannot be determined

## 64.8 regression\_method\_of\_moments

fit linear function  $\|a \ b \ x = y\|_{L2}$  by the method of moments  
 $y + \epsilon = \alpha + \beta x$

## 64.9 robustlinreg

fit a linear function by splitting the x-values at their median  
 $(\text{med}(y_{\text{left}}) - \text{med}(y_{\text{right}})) / (\text{med}(x_{\text{left}}) - \text{med}(x_{\text{right}}))$   
this approach performs poorly compared to the theil-senn operator

## 64.10 theil2

Theil senn-estimator for two dimensions (glm)

## 64.11 theil\_generalised

generalization of the Theil-Senn operator to higher dimensions,  
for arbitrary functions such as polynomials and multivariate  
regression  
either higher order polynomials or glm  
c.f. "On theil's fitting method", Pegoraro, 1991

## 64.12 total\_least\_squares

total least squares

## 64.13 weighted\_median\_regression

weighted median regression  
c.f. Scholz, 1978

# 65 set-theory

## 65.1 issubset

test if set B is subset of A in  $O(n)$ -runtime

A : first set  
B : second set  
P : set of primes (auxiliary)

# 66 signal-processing

## 66.1 acf\_effective\_sample\_size

effective sample size from acf

## 66.2 acf\_genton

autocorrelation function

### 66.3 acfar1

Autocorrelation function of the finite AR1 process

```
a_k = 1/(n-k)sum x_ix_{i+1} + (x_i + x_{i+k})mu + mu^2
      = r^k + 1/n sum_{ij} + 1/n
      pause
```

### 66.4 acfar1\_2

autocorrelation of the ar1 process

### 66.5 acfar2

impulse response of the ar2 process

### 66.6 acfar2\_2

autocorrelation of the ar2 process  
 $X_i + a_1 X_{i-1} + a_2 X_{i-2} = 0$

### 66.7 ar1\_cutoff\_frequency

### 66.8 ar1\_effective\_sample\_size

effective sample size correction for autocorrelated series

### 66.9 ar1\_mse\_mu\_single\_sample

standard error of a single sample of an ar1 correlated process

### 66.10 ar1\_mse\_pop

variance of the population mean of a single realisation around zero

$$E[(\mu_N - 0)^2] = E[\mu_N^2]$$

### 66.11 ar1\_mse\_range

mean standard error of the mean of a range of values taken from an ar1 process

### 66.12 ar1\_spectrum

spectrum of the ar1 process

### 66.13 ar1\_to\_tikhonov

convert ar1 correlation to tikhonovs lambda

### 66.14 ar1\_var\_factor

variance correction factor for an autocorrelated finite process

n : [1 .. inf] population size

m : [1 .. n] samples size

rho : [ -1 < rho < 1 (for convergence) ] correlation of samples

### 66.15 ar1\_var\_factor\_

variance of an autocorrelated finite process

### 66.16 ar1\_var\_range2

variance of sub sample starting at the end of the series

from the finite length first order autocorrelated process

$$s2 = 1/m^2 \sum_i^m \sum_j^m \rho^{|i-j|}$$

### 66.17 ar1delay

approximate acf by the ar1 process  
acf: autocovariance or autocorrelation function  
nf : skip first samples (for mixed geometric-arithmetic series (ARMA))

### 66.18 ar1delay\_old

autocorrelation of the residual

### 66.19 ar2conv

coefficients of the ar2 process determined from the two leading correlations  
of the acf [1,r1,r2,...]

### 66.20 ar2dof

effective samples size for the ar2 process

### 66.21 ar2param

ar2 parameter estimation from first two terms of acf  
acf = [1 a1 a2 ...]

### 66.22 asymwin

creates asymmetrical filter windows  
filter will always have negative weights

### 66.23 autocorr\_fft

autocorrelation function

## 66.24 bandpass

bandpass filter

## 66.25 bandpass2

bandpass filter

## 66.26 bartlett

Effective sample size factor for bartlett window  
c.f. thiebaux  
c.f spectral analysis-jenkins, eq. (6.3.27)  
 $c = acf$   
note: results seams always to be 1 tac too low  
T : reduction factor for dof  
for ar1 with  $a = \rho^k = \exp(-k/L)$ ,  $T = 2L$

## 66.27 bartlett\_spectrogram

bartlet spectrogramm  
TODO sliding window

## 66.28 bin1d

bin values of v sampled at x into bins bounded by "edges"  
apply function v to it

## 66.29 bin2d

bin values of V sampled at X and Y into the grid structured grid ex  
,ey  
apply function func to all wvalues in the bin  
func = mean : default  
func = sum : non-normalized frequency histogram in 2D

### **66.30 binormrnd**

generate two correlated normally distributed vectors

### **66.31 conv1\_man**

convolutions with padding

### **66.32 conv2\_man**

convolution in 2d

### **66.33 conv2z**

### **66.34 conv30**

convolve with rectangular window of length n  
circular boundaries

### **66.35 conv\_**

convolution of a with b

### **66.36 conv\_centered**

convolve x with filter window f  
when length of f is even, this guarantees a symmetric result (no  
off by on  
displacement) by making the length of f odd at first

### **66.37 convz**



### 66.38 cosexpdelay

### 66.39 csmooth

smooth recursively with  $[1,2,1]/4$  kernel  
function `x = csmooth(x,n,p,circ)`

### 66.40 daniell\_window

Daniell window for smoothing the power spectrum  
c.f. Daniell 1946  
Bloomfield 2000  
meko 2015

### 66.41 danielle\_window

danielle fourier window

### 66.42 db2neper

convert decibel to neper

### 66.43 db2power

power ratio from db

### 66.44 derive\_danielle\_weight

### 66.45 derive\_limit\_0\_acfar

#### 66.46 detect\_peak

detect peaks in a vector  
requires function value to fall to  $p \cdot \max$  before new value is  
allowed

#### 66.47 digital\_low\_pass\_filter

design coefficients of a low pass filter with specified cut of  
frequency  
and sampling period  
analogue low pass with pole at  $s = -\omega_c = 1/\tau = 1/RC$   
 $H_a = \tau / (\tau + s) = 1 / (1 + \omega_c \cdot s)$

#### 66.48 doublesum\_ij

double sum of  $r^i$

#### 66.49 effective\_sample\_size\_to\_ar1

convert effective sample size to ar1 correlation

#### 66.50 flt\_hodges\_lehman

#### 66.51 filter1

filter along one dimension

#### 66.52 filter2

filter columns of  $x$  (matlab does only support vector input)

### 66.53 filter\_

invalidate values that exceed n-times the robust standard deviation

### 66.54 filteriir

filter adcp t-n data over time

v : nz,nt : values to be filtered

H : nt,1 : depth of ensemble

last : nt,1 : last bin above bottom that can be sampled without  
side lobe interference

nf : scalar : number of reweighted iterations

when samples

- distance to bed is reference (advantageous for near-bed suspended  
transport)

TODO for wash load: distance to surface is more relevant  
interpolate depending on z

when depth changes, neighbouring indices do not correspond to same  
relative position in the water column

relative position in the column (s-coordinate) smoothes values

near the bed: absolute distance to bed is chosen

near surface: absolute distance to surface is chosen

-> cubic transformation of index

faster and avoid aliasing (smoothing along z)

resample ensemble to same number of bins in S -> filter ->

resample back

use nonlinear transform z-s coordinates

-> resampling has to be local (Hi -> H-filtered)

filtered profile coordinates to sample coordinates

zf -> zi (special transform)

corresponding indices and fractions

filtration step (update of hf and vf)

sample coordinates to updated profile coordinates

(the inverse step is actually not necessary)

write filtered value

### 66.55 filterp

### **66.56 filterp1**

fir filter with some fancy extras

### **66.57 filterstd**

### **66.58 firls\_man**

design finite impulse response filter by the least squares method

### **66.59 flattopwin**

the flat top window

### **66.60 frequency\_response\_boxcar**

frequency response of a boxcar filter

### **66.61 freqz\_boxcar**

frequency response of a boxcar filter

### **66.62 gaussfilt1**

filter data series with a gaussian window

### **66.63 hanchangewin**

hanning window for change point detection

#### **66.64 hanchangewin2**

nanning window for chage point detection

#### **66.65 hanwin**

hanning filter window

#### **66.66 hanwin\_**

hanning filter window

#### **66.67 highpass**

high pass filter

#### **66.68 kaiserwin**

kaiser filter window

#### **66.69 kalman**

Kalman filter

#### **66.70 lanczoswin**

Lanczos window

#### **66.71 last**

lake tail, but for matrices

## **66.72 lowpass**

low pass filter

## **66.73 lowpass2**

design low pass filter with cutoff-frequency f1

## **66.74 lowpass\_iir**

iir-low pass

## **66.75 lowpass\_iir\_symmetric**

two-sided iir low pass filter (for symmetry)

## **66.76 lowpassfilter2**

low-pass filter of data

## **66.77 maxfilt1**

## **66.78 meanfilt1**

moving average filter with special treatment of the boundaries

## **66.79 medfilt1\_man**

moving median filter, supports columnwise operation

## 66.80 medfilt1\_man2

moving median filter with special treatment of boundaries

## 66.81 medfilt1\_padded

median filter with padding

## 66.82 medfilt1\_reduced

median filter with padding

## 66.83 mid\_term\_single\_sample

variance of single sample, mid term

## 66.84 minfilt1

## 66.85 mu2ar1

error variance of the mean of the finite length ar1 process

$(\mu)^2 = (\sum \epsilon_i)^2 = \sum_i \sum_j \epsilon_i \epsilon_j = \sum_{ii}(\rho, n)/n^2$   
this has the limit  $s^2$  for  $\rho \rightarrow 1$

## 66.86 mysmooth

## 66.87 nanautocorr

autocorrelation with nan-values

#### **66.88 nanmedfilt1**

medfilt1, skipping nans

#### **66.89 neper2db**

convert neper to db

#### **66.90 peaks\_man**

peaks of a periodogram

#### **66.91 polyfilt1**

polynomial filter,  
can be achieved by iteratively processing the data with  
a mean (zero-order) filter

#### **66.92 qmedfilt1**

medfilt1, after fitting a quadratic polynomial

#### **66.93 randar1**

generate random ar1 process  
e1 = randar1(sigma,p,n,m)

#### **66.94 randar1\_dual**

draw random variables of two correlated ar1 processes

#### **66.95 randar2**

generate ar2 process



### 66.96 randarp

randomly generate the instance of an ar-p process

### 66.97 range\_window

range of values within a certain range of indices (window)

### 66.98 rectwin

rectangular window

### 66.99 recursive\_sum

### 66.100 select\_range

### 66.101 smooth1d\_parametric

smooth position of  $p_0=x_0, y_0$  between  $p_1=x_1, y_1$  and  $p_2=x_2, y_2$ ,  
so that distance to  $p_1$  and  $p_2$  becomes equal  
and the chord length remains the same

### 66.102 smooth2

smooth vectos of  $X$

### 66.103 smooth\_man

#### 66.104 smooth\_parametric

smooth a parametric function given in x-y coordinates  
matvec2x2(R,[dxc;dyc])

#### 66.105 smooth\_parametric2

parametrically smooth the curve

#### 66.106 smooth\_with\_splines

#### 66.107 smoothfft

filter with fast fourier transform

#### 66.108 spectrogram

spectrogram

#### 66.109 std\_window

moving block standard deviation

#### 66.110 sum\_i\_lag

sum of ar1 matrix with lag  
 $\sum_{i=1}^n \rho^{|i-k|}$

#### 66.111 sum\_ii

sum of ar1 matrix  
 $\sum_{i=1}^n \sum_{j=1}^n \rho^{|i-j|}$   
this is for the variance, take square root for the standard  
deviation factor

66.112 sum\_ii\_

66.113 sum\_ij

sum of ar1 matrix  
 $\sum_{i=1}^n \sum_{j=1}^m r^{|i-j|}$

66.114 sum\_ij\_

66.115 sum\_ij\_partial\_

66.116 sum\_multivar

sum of matrix entries of bivariate ar1 process

66.117 test\_acfar1

66.118 test\_acfar1\_2

66.119 test\_acfar1\_3

66.120 test\_acfar1\_4

66.121 test\_acfar2

66.122 test\_ar1\_var\_factor

66.123 test\_ar1\_var\_factor\_2

66.124 test\_ar1\_var\_mu\_single\_sample

66.125 test\_ar1\_var\_pop

66.126 test\_ar1\_var\_pop\_1

66.127 test\_ar1delay

66.128 test\_bivariate\_covariance\_term

66.129 test\_convexity

66.130 test\_lanczoswin

66.131 test\_madcorr

66.132 test\_randar1

66.133 test\_randar1\_multivariate

66.134 test\_randar2

66.135 test\_sum\_ij

66.136 test\_sum\_multivar

66.137 test\_trifilt1

66.138 test\_wautocorr

66.139 test\_wavelet\_transform

66.140 test\_wordfilt

**66.141**   `test_xar1_mid` term

**66.142**   `tikhonov_to_ar1`

convert coefficient of the tikhonov regularization to correlation  
of the ar1 process

**66.143**   `trapwin`

trapezoidal filter window

**66.144**   `trifilt1`

filter with triangular window

**66.145**   `triwin`

triangular filter window

**66.146**   `triwin2`

triangular filter window

**66.147**   `varar1`

error variance of a single sample of a finite length ar1 process  
with respect to the mean, averaged over the population

**66.148**   `welch_spectrogram`

welch spectrogram

#### 66.149 wfilt

filter with window

#### 66.150 winbandpass

filter with bandpass

#### 66.151 window\_make\_odd

#### 66.152 winfilt0

filter with window

#### 66.153 winlength

window length for desired cutoff frequency  
power at  $f_c$  is halved  
 $H(wf) = 1/\sqrt{2} H(f)$   
if the filter window were used as a low pass filter  
note: the user should prefer a windowed ideal low pass filter  
TODO, relate this to DOF

#### 66.154 wmeanfilt

mean filter with window

#### 66.155 wmedfilt

median filter with window

## **66.156 wordfilt**

weighted order filter

## **66.157 wordfilt\_edgeworth**

weighed order filter

## **66.158 xar1**

## **66.159 xcorr\_man**

cross correlation of two sampled ar1 processes

# **67 sorting**

## **67.1 sort2**

sort two numbers

## **67.2 sort2d**

sort elements of matrix in X  
returns row and column index of sorted values

# **68 special-functions**

## **68.1 bessell\_sphere**

spherical Bessel function of the first kind



## 68.2 digamma\_man

## 68.3 hankel\_sphere

spherical Hankel function for the far field (incident plane wave)  
first kind

## 68.4 hermite

probabilistic's hermite polynomial by recurrence relation

input :  
n : order  
x : value

output:  
f :  $H_n(x)$   
df :  $d/dx H_n(x)$

## 68.5 legendre\_man

legendre polynomials

## 68.6 neumann\_sphere

spherical Neumann function  
Bessel function of the second kind

# 69 statistics

## 69.1 atan\_s2

stadard deviation of the arcus tangens by means of taylor expansion

## **69.2   beta\_mode\_to\_parameter**

transform modes (mean and sd) to params of the beta function

## **69.3   coefficient\_of\_determination**

## **69.4   conditional\_expectation\_normal**

## **69.5   correlation\_confidence\_pearson**

confidence intervals of the correlation coefficient  
c.f. Fischer 1921

# **70   statistics/distributions**

## **70.1   PDF**

class for quasi-distributions from a set of sampling points

## **70.2   binorm\_separation\_coefficient**

separation coefficient of a bimodal normal distribution

## **70.3   binormcdf**

bio-modal gaussian distribution

## **70.4   binormfit**

fit sum of two normal distribution to a histogram

## 70.5 binormpdf

## 70.6 edgeworth\_cdf

edgeworth expansion of an unknown cumulative distribution  
with mean mu, standard deviation sigma, and third and fourth  
cumulants  
c.f. Rao 2010

## 70.7 edgeworth\_pdf

probability density of and unknown distribution  
with mean mu, standard deviation sigma, and third and fourth  
cumulants  
c.f. Rao 2010

## 70.8 logn\_mode2param

transform modes (mu,sd) to parameters of the log normal  
distribution

## 70.9 logn\_param2mode

transform parameters to mode (mu, sd) for the log normal  
distribution

## 70.10 lognpdf\_

log normal distribution called by modes rather than parameters

## 70.11 pdfsample

pdf from sample distribution  
Note: better use kernel density estimates

## 70.12 `t2cdf`

Hotelling's T-squared cumulative distribution

## 70.13 `t2inv`

inverse of Hotelling's T-squared cumulative distribution

# 71 `statistics`

## 71.1 `example_standard_error_of_sample_quantiles`

## 71.2 `f_var_finite`

reduction of variance when sampling from a finite population  
without replacement

## 71.3 `gamma_mode_to_parameter`

transform modes ( $\mu, sd$ ) to parameters of the gamma distribution

## 71.4 `gaussfit3`

## 71.5 `gaussfit_quantile`

## 71.6 `geoserr`

## 71.7 geostd

## 71.8 hodges\_lehmann\_correlation

hodges\_lehmann correlatoon coefficient  
c.f. Shamos 1976  
c.f. Bickel and Lehmann 1976  
c.f. rousseeuw 1993  
c.f. Shevlyakov 2011

## 71.9 hodges\_lehmann\_dispersion

dispersion determined by the hodges lehman method  
asymptotic efficiency of dispersion estimates:  
standard deviation:  $E(s - \hat{s})/s = 2/\sqrt{2n} \sim 0.707/\sqrt{n}$   
(100%)  
hodges lehmann dispersion  $E(s - \hat{s})/s = (\pi/3)^2 / (\sqrt{2n}) \sim$   
 $0.775/\sqrt{n}$  (91%)  
mad  $E(s - \hat{s})/s \sim 1.17 s/\sqrt{n}$   
(60%)  
c.f. Shamos 1976  
c.f. Bickel and Lehmann 1976  
c.f. rousseeuw 1993  
nb: rousseeuw uses the 25th percentile, which is more efficient for  
small sample sizes

## 72 statistics/information-theory

### 72.1 akaike\_information\_criterion

akaike information criterion  
  
serr : rmse of model prediction  
n : effective sample size  
k : number of parameters  
  
c.f. akaike (1974)  
c.f. sugiura 1978

## **72.2   bayesian\_information\_criterion**

bayesian information criterion

## **73   statistics**

### **73.1   kurtncdf**

### **73.2   kurtnpdf**

### **73.3   kurtosis\_bias\_corrected**

bias corrected kurtosis

### **73.4   limit**

limit a by lower and upper bound

### **73.5   logfactorial**

approximate log of the factorial

### **73.6   loglogpdf**

### **73.7   lognfit\_quantile**

## 73.8 logskewcdf

## 73.9 logskewpdf

# 74 statistics/logu

## 74.1 lambertw\_numeric

lambert-w function

## 74.2 logtrialtcdf

pdf of a logarithmic triangular distribution

## 74.3 logtrialtinv

inverse of the logarithmic triangular distribution

$$= (d F \log(a) \log(b) + a \log(b) - b \log(a) - d F \log(a) \log(c) - a \log(c) + d F \log(b) \log(c) + b \log(c) - d F \log^2(b)) / ((\log(a) - \log(b)) W((a^{-1/(\log(a) - \log(b))}) (b^{-\log(c)/\log(a) - 1/\log(a)}) c)^{-\log(a)/(\log(a) - \log(b))}) (-d F \log^2(b) + a \log(b) + d F \log(a) \log(b) + d F \log(c) \log(b) - b \log(a) - a \log(c) + b \log(c) - d F \log(a) \log(c))) / (\log(a) - \log(b)))$$
$$x = (d F \log(a) \log(b) + a \log(b) - b \log(a) - d F \log(a) \log(c) - a \log(c) + d F \log(b) \log(c) + b \log(c) - d F \log^2(b)) / ((\log(a) - \log(b)) W((a^{-1/(\log(a) - \log(b))}) (b^{-\log(c)/\log(a) - 1/\log(a)}) c)^{-\log(a)/(\log(a) - \log(b))}) (-d F \log^2(b) + a \log(b) + d F \log(a) \log(b) + d F \log(c) \log(b) - b \log(a) - a \log(c) + b \log(c) - d F \log(a) \log(c))) / (\log(a) - \log(b)))$$

## 74.4 logtrialtmean

mean of the logarithmic triangular distribution

## 74.5 logtrialtpdf

density of the logarithmic triangular distribution

## 74.6 logtrialtrnd

## 74.7 logtricdf

cumulative distribution of the logarithmic triangular distribution

## 74.8 logtriinv

inverse of the logarithmic triangular distribution

## 74.9 logtrimean

mean of the logarithmic triangular distribution

## 74.10 logtripdf

probability density of the logarithmic triangular distribution

## 74.11 logtirnd

## 74.12 logucdf

probability density of the logarithmic uniform distribution



### 74.13 logucm

central moments of the log-uniform distribution

### 74.14 loguinv

inverse of the log-uniform distribution

### 74.15 logumean

mean of the log-uniform distribution

### 74.16 logupdf

pdf of the log uniform distribution

### 74.17 logurnd

random numbers following a log-uniform distribution

### 74.18 loguvar

variance of the log-uniform distribution

### 74.19 medlogu

median of the log-uniform distribution

### 74.20 test\_logurnd

### 74.21 `tricdf`

cumulative distribution of the log-triangular distribution

### 74.22 `triinv`

inverse of the triangular distribution

### 74.23 `trimedian`

median of the triangular distribution

### 74.24 `tripdf`

probability density of the triangular distribution

### 74.25 `trirnd`

random numbers of the triangular distribution

## 75 statistics

### 75.1 `max_exprnd`

### 75.2 `maxnnormals`

expected maximum of n normal variables  
c.f. Wolperts  
this is the median, not the mean of the maximum!  
see median of gumbel

### 75.3 mean\_generalized\_gampdf

### 75.4 midrange

mid range of columns of X

### 75.5 minavg

solution of the minimum variance problem  
minimise the variance of the weighted sum of n-independent  
random variables with equal mean and individual variance

### 75.6 mode\_man

## 76 statistics/moment-statistics

### 76.1 autocorr\_man3

autocorrelation of the columns of X

### 76.2 autocorr\_man4

autocorrelation for x if x is a vector, or individually for the  
columns of x if x is a matrix

c.f. box jenkins 2008 eq. 2.1.12

Note that it is faster to compute the acf in frequency space  
as done in the matlab internal function

### 76.3 autocorr\_man5

autocorrellation of the columns of X

## 76.4 blockserr

estimate the standard error of potentially sequentially correlated data  
by blocking  
block length should be sufficiently larger than correlation length  
and sufficiently smaller than data length  
this uses a sliding block approach, which reduces the variation of the error estimate

## 76.5 comoment

non-central higher order moments of the multivariate normal distribution

c.f. Moments and cumulants of the multivariate real and complex Gaussian distributions

note : there seem to be some typos in the original paper,  
for  $x^4 c_{ii}^2$ , the square seems to be missing  
 $\mu$  :  $n \times 1$  mean vector  
 $C$  :  $n \times n$  covariance matrix  
 $k$  :  $n \times 1$  powers of variables in moments

## 76.6 corr\_man

correlation of two vectors

## 76.7 cov\_man

covariance matrix of two vectors

## 76.8 dof

minimum number of support points  
for a polynomial of degree order in  $\dim$  dimensions

## 76.9 edgeworth\_quantile

inverse edgeworth expansion  
c.f. cornis fisher 1937  
c.f. Rao 2010  
c.f. 2.50 in hall  
CHERNOZHUKOV 3.3

## 76.10 effective\_sample\_size

effective sample size of the weighted mean of uncorrelated data  
c.f. Kish

## 76.11 f\_correlation

correction factor for standard error of the mean of  $n$  ar1-  
correlated iid samples

## 76.12 f\_finite

reduction factor of standard error for sampling from a finite  
distribution  
without replacement

## 76.13 lmean

mean of  $x.^l$ , not of abs

## 76.14 lmoment

l-moment of vector  $x$

## 76.15 maskmean

mean of the masked values of  $X$

#### 76.16 masknanmean

#### 76.17 mean1

mean of x

#### 76.18 mean\_man

mean and standard error of X

#### 76.19 mse

mean squared error of residual vector res  
this is de-facto the std for an unbiased residual

#### 76.20 nanautocorr\_man1

autocorrelation of a vector with nan-values

#### 76.21 nanautocorr\_man2

autocorrelation of a vector with nan-values

#### 76.22 nanautocorr\_man4

compute autocorrelation for x if x is a vector, or individually  
for the  
columns of x if x is a matrix  
box jenkins 2008 eq. 2.1.12  
TODO nan is problematic!  
Note that it is faster to compute the acf in frequency space  
as done in the matlab internal function

### 76.23 `nancorr`

(co)-correlation matrix when samples a NaN

### 76.24 `nancumsum`

cumulative sum, setting nan values to zero

### 76.25 `nanlmean`

mean of the l-th power of the absolute value of x

### 76.26 `nanr2`

coefficient of determination when samples are invalid

### 76.27 `nanrms`

root mean square value when sample contains nan-values

### 76.28 `nanrmse`

root mean square error from vector of residuals  
this is de-facto the std for an unbiased residual

### 76.29 `nanserr`

standard error of x with respect to mean when x contains nan values

### 76.30 `nanwmean`

weighted mean  
 $\min_x \sum w (x - \mu)^2 \Rightarrow \mu = \sum(wx) / \sum(w)$   
varargin can be dim  
function [mu serr] = nanwmean(w,x)

### 76.31 nanwstd

weighed standard deviation

### 76.32 nanwvar

weighted variance of columns, corrected for degrees of freedom (bessel)

$$s^2 = \text{sum}(w*(x-\text{sum}(wx)/\text{sum}(w))^2)/\text{sum}(w)$$

### 76.33 nanxcorr

### 76.34 pearson

pearson correlation coefficient

### 76.35 pearson\_to\_kendall

conversion of pearson to kendall correlation coefficient  
c.f. Kruskal 1958

### 76.36 pool\_samples

pooled mean and standard deviation of several groups of different size, mean and standard deviation

### 76.37 qmean

trimmed mean

### 76.38 range\_mean



### 76.39 rmse\_

root mean square error computed from a residual vector  
this is de-facto the std for an unbiased residual

### 76.40 serr

standard error of the mean of a set of uncorrelated samples

### 76.41 serr1

### 76.42 test\_qskew

### 76.43 test\_qstd\_qskew\_optimal\_p

### 76.44 wautocorr

autocorrelation for x if x is a vector, or individually for the  
columns of x if x is a matrix  
samples can be weighted

c.f. box jenkins 2008 eq. 2.1.12

c.f. autocorr\_man4

Note that it is faster to compute the acf in frequency space  
as done in the matlab internal function

### 76.45 wcorr

correlation of two vectors when samples are weighted

#### 76.46 wcov

covariance of two vectors when samples are weighted

#### 76.47 wdof

effective degrees of freedom for weighted samples

#### 76.48 wkurt

kurtosis with weighted samples

#### 76.49 wmean

weighted mean

$\min_x \sum w (x - \mu)^2 \Rightarrow \mu = \sum(wx) / \sum(w)$

varargin can be dim

function [mu serr] = wmean(w,x)

#### 76.50 wrms

weighted root mean square error

#### 76.51 wserr

weighted root mean square error

#### 76.52 wskew

skewness of a weighted set of samples

## 76.53 wstd

weighed standard deviation

## 76.54 wvar

weighted variance of columns, corrected for degrees of freedom (bessel)  
variance of the weighted sample mean of samples with same mean (but not necessarily same variance)  
 $s^2 = \text{sum } (w^2(x - \text{sum}(wx))^2)$   
  
 $s2\_mu$  : error of mean,  $s2\_mu$  : sd of prediction

## 77 statistics

### 77.1 nangeomean

### 77.2 nangeostd

geometric standard deviation ignoring nan-values

## 78 statistics/nonparametric-statistics

### 78.1 kernel1d

$X$  : ouput x axis bins  
 $xi$  : samples along x  
 $m$  : number of bins in  $X$   
 $fun$  : kernel function  
 $pdf$  : propability density of  $xi$

### 78.2 kernel2d

kernel density estimate in two dimensions

## 79 statistics

### 79.1 normmoment

expected norm of  $x.^n$ , when values  $x$  in  $x$  are iid normal with  $\mu$   
and  $\sigma$

### 79.2 normpdf2

pdf of the bivariate normal distribution

## 80 statistics/order-statistics

### 80.1 hodges\_lehmann\_location

hodges lehman location estimator

Asymptotic rms efficiency of location estimate:

mean:	$1 s/\sqrt{n}$
hodges lehman:	$\sqrt{\pi/3} * s \sim 1.0233 s/\sqrt{n}$
median:	$\pi/2 s/\sqrt{n} \sim 1.25 s / \sqrt{n}$

### 80.2 kendall

kendall correlation coefficient

### 80.3 kendall\_to\_pearson

convert kendall rank correlation coefficient to the person product  
moment  
correlation coefficient

c.f. Kruska, 1985

### 80.4 mad2sd

transform median absolute deviation to standard deviation  
for normal distributed values

## 80.5 madcorr

proxy correlation by median absolute deviation

## 80.6 median2\_holder

## 80.7 median\_ci

median and its confidence intervals under assumption of normality  
 $se\_me = \sqrt{1/2 \pi} \cdot 1.25331 \cdot sd/\sqrt{n}$

## 80.8 median\_man

median and confidence intervals  
c is a P value for the confidence interval,  
default is 0.95 (2-sigma)  
median of the columns of X

## 80.9 mediani

index of median, if median is not unique, any of the values is  
chosen

## 80.10 nanmadcorr

proxy correlation by median absolute deviation

## 80.11 nanwmedian

weighted median, skips nan-values

## 80.12 nanwquantile

weighted quantile, skips nan values

## 80.13 oja\_median

two dimensional oja median

note: the multivariate median is not unique

oja 1983, for extension to multivariate function, see chaudhri

## 80.14 qkurtosis

kurtosis computed for quantiles

Note : this is a measurement of shape-tailedness and yields the same value for the

normal distribution as "kurtosis"

However, this is a separate statistic and hence requires different

methods for calculating P-values and hypothesis testing

## 80.15 qmoments

moments estimated from quantiles

## 80.16 qskew

skewness estimated from quantiles

Note : this is a measurement of shape-symmetry and yields the same value for the

skew-normal distribution as "skewness"

However, this is an own statistic and hence requires different

methods for calculating P-values and hypothesis testing

### 80.17 qskewq

skewness estimated by quantiles

### 80.18 qstdq

proxy standard deviation determined by quantiles

### 80.19 quantile1\_optimisation

### 80.20 quantile2\_breckling

quantile regression

### 80.21 quantile2\_chaudhuri

quantile regression

### 80.22 quantile2\_projected

quantile in two dimensions

### 80.23 quantile2\_projected2

spatial quantile for chosen direction

### 80.24 quantile\_envelope

## 80.25 quantile\_regression\_simple

simple quantile regression

## 80.26 ranking

ranking for spearman statistics

## 80.27 spatial\_median

c.f. Oja 2008

is this the same as the oja simplex median (c.f. small 1990)?

## 80.28 spatial\_quantile

spatial quantile

## 80.29 spatial\_quantile2

spatial quantile

## 80.30 spatial\_quantile3

spatial quantile

## 80.31 spatial\_rank

unsigned rank

## 80.32 spatial\_sign

spatial sign



### 80.33 `spatial_signed_rank`

signed rank

Note: this is only a true rank if X is normal with zero mean,  
arbitrary variance

### 80.34 `spearman`

spearman's product moment coefficient

### 80.35 `spearman_rank`

### 80.36 `spearman_to_pearson`

conversion of spearman rank to person product moment correlation  
coefficient

### 80.37 `wmedian`

weighted median

### 80.38 `wquantile`

weighted quantile

## 81 `statistics`

### 81.1 `qstd`

### 81.2 `quantile_extrap`

## **82 statistics/random-number-generation**

### **82.1 laplacernd**

random number of laplace distribution

### **82.2 randc**

correlate to correlated standard normally distributed vectors

### **82.3 skewness2param**

### **82.4 skewpdf\_central\_moments**

### **82.5 skewrnd**

random numbers of the skew normal distribution

### **82.6 skewrnd2**

random numbers of the skew normal distribution

## **83 statistics**

### **83.1 range**

mid range

### **83.2 resample\_with\_replacement**

## 84 statistics/resampling-statistics/@Jackknife

### 84.1 Jackknife

class for leave out 1 (delete 1) Jackknife estimates

note 1 : the 1-delete jackknife does not yield consistend estimates  
for all functions,  
in particular it will perform poorly on robust estimation  
functions  
this is overcome by the d-delete jackknife, where d has to  
exceed the breakdown point  
of the estimating function, for example  $\sqrt{n}$  for the  
median  
as this leads to unreasonably large number of repetitions,  
bootstrap  
is recommended for large sample cases (or blocking for  
sequential data)

note 2 : as a linearisation, jackknife underestimates the error  
variance in case of  
dependence in the data

note 3 : studentisation and the leave out 1 jackknife are related

note 4 : the double 1 sample jackknife performs iferior to the d1  
jackknife

### 84.2 estimated\_STATIC

jackknife estimate of mean, bias and standard error  
theta0 : estimate from all samples  
thetad : set of estimates obtained by leaving out one data point  
each  
last dimension of theta is assumed to be the jackknife  
dimension

### 84.3 matrix1\_STATIC

matrix of estimation for leaving out two samples at a time

### 84.4 matrix2

matrix of estimations for jacknive with two samples left out

## 85 statistics/resampling-statistics

### 85.1 block\_jackknife

### 85.2 jackknife\_moments

moments determined by the jackknife

```
func : function of interest on the samples (e.g. mean)
A    : parameter matrix
      columns : parameters
      rows    : samples of the parameter sets
d    : number of samples left out
```

### 85.3 moving\_block\_jackknife

blocked Jackknife for autocorrelated data  
sliding block, statistically more efficient but computationally  
expensive  
note, number of blocks must be sufficiently large  $h \sim \sqrt{n}$ ?  $\ll n$

### 85.4 randblockterr

standard error of sequentially correlated data by blocking  
block length should be sufficiently larger than correlation length  
and sufficiently smaller than data length  
this uses a sliding block approach, which reduces the variation of  
the error estimate  
TODO this does not work, randomly picking samples does not reveal  
the correlation

### 85.5 resample

resample a vector and apply function to it

TODO, should be with replacement

```
n : number of samples
```

m : number of subsamples  
cx : maximum number of combinations

## 86 statistics

### 86.1 scale\_quantile\_sd

scale factor for the standard deviation  
of the asymptotic distribution of sample quantiles  
(for normal distribution)  
see cadwell, 1952

### 86.2 sd\_sample\_quantiles

### 86.3 skewpdf

skew-normal distribution  
c.f. Azzalini 1985

### 86.4 test\_mean\_generalized\_gampdf

### 86.5 trimmed\_mean

trimmed mean

### 86.6 ttest2\_man

two-sample t-test  
here posix return value standard: h = 0 accepted, h = 1 failed  
note: the matlab logic is inverse : h = 1 accepted, h = 0 failed  
two sided univariate t-test

## 86.7 ttest\_man

two-sample t-test  
unequal sample size  
equal variance

## 86.8 ttest\_paired

paired t-test  
unequal sample size  
equal variance  
more powerfull than unpaired test, as long as correlation between  
x1 and x2 > 0

## 86.9 wgeomean

weighted geometric mean  
function mu = wgeomean(w,x)

## 86.10 wgeovar

variance of the weighted geometric mean

## 86.11 wharmean

weighted harmonic mean

## 86.12 wharstd

## 86.13 wharvar

## 87 mathematics

mathematical functions of various kind

### 87.1 ternary\_diagram

## 88 test/master

### 88.1 dat\_test\_lanczos\_3d\_k\_20\_n\_40

### 88.2 poisson2d\_blk

### 88.3 qr\_implicit\_givens\_2

### 88.4 spectral\_derivative\_2d

### 88.5 test\_2d\_eigensolver\_hydrogen

### 88.6 test\_2d\_refine

### 88.7 test\_3d\_eigensolver\_hydrogen

88.8 test\_FEM

88.9 test\_Mesh\_3d

88.10 test\_arnoldi

88.11 test\_arpackc

88.12 test\_assemble

88.13 test\_assembly\_performance

88.14 test\_bc\_one\_sided

88.15 test\_compare\_solvers

88.16 test\_complete

88.17 test\_convergence



88.18 test\_convergence\_b

88.19 test\_df\_2d

88.20 test\_eig\_algs

88.21 test\_eig\_inverse

88.22 test\_eigs\_lanczos

88.23 test\_eigs\_lanczos\_1

88.24 test\_eigs\_lanczos\_2

88.25 test\_eigs\_lanczos\_performance

88.26 test\_fdm

88.27 test\_fdm\_d\_vargrid

88.28 test\_fdm\_spectral

88.29 test\_fem

88.30 test\_fem\_1d

88.31 test\_fem\_1d\_higher\_order

88.32 test\_fem\_2d\_adaptive

88.33 test\_fem\_2d\_higher\_order

88.34 test\_fem\_3d\_higher\_order

88.35 test\_fem\_3d\_refine

88.36 test\_fem\_b

88.37 test\_fem\_derivative

88.38 test\_fem\_quadrature

88.39 test\_final

88.40 test\_fix\_substitution

88.41 test\_forward

88.42 test\_get\_sparse\_arrays

88.43 test\_harmonic\_oscillator

88.44 test\_high\_order\_fdm\_periodic\_bc

88.45 test\_hydrogen\_wf

88.46 test\_ichol

88.47 test\_interpolation

88.48 test\_inverse\_problem

88.49 test\_it\_vs\_exact

88.50 test\_jama

88.51 test\_jd

88.52 test\_jdqz

88.53 test\_lanczos\_2

88.54 test\_lanczos\_biorthogonal

88.55 test\_laplacian

88.56 test\_laplacian\_non\_uniform

88.57 test\_laplacian\_simple

88.58 test\_mesh\_2d\_uniform

88.59 test\_mesh\_2d\_uniform\_2

88.60 test\_mesh\_circle

88.61 test\_mesh\_generation

88.62 test\_mesh\_interpolate

88.63 test\_mg

88.64 test\_minres\_recycle

88.65 test\_multigrid

88.66 test\_nc

88.67 test\_nonuniform\_symmetric

88.68 test\_pde

88.69 test\_permutation

88.70 test\_poison\_fem

88.71 test\_polar

88.72 test\_potential

88.73 test\_powers

88.74 test\_precondition

88.75 test\_project\_rectangle

88.76 test\_qr

88.77 test\_quantum\_well

88.78 test\_radial\_adaptive

88.79 test\_radial\_confinement

88.80 test\_radial\_fixes

88.81 test\_refine\_2d

88.82 test\_refine\_2d\_b

88.83 test\_refine\_3d

88.84 test\_refine\_structural

88.85 test\_regularisation

88.86 test\_round\_off

88.87 test\_schrödinger\_potentials

88.88 test\_uniform\_mesh

88.89 test\_vargrid

89 test

89.1 test\_gaussfit3

89.2 test\_geoserr

89.3 test\_lognfit\_quantile

89.4 test\_max\_normal

89.5 test\_mtimes3x3

90 mathematics

mathematical functions of various kind

90.1 vanderd\_2d



## 91 wavelet

### 91.1 continuous\_wavelet\_transform

continuous wavelet transform  
follows "The Illustrated Wavelet Transform Handbook: Introductory  
Theory and ..."

### 91.2 cwt\_man

continuous fourier transform  
as of time of implmentation, the matlab interal cwt is affected by  
serious round-off errors and has issues with the scaling,  
which is not the case here

### 91.3 example\_wavelets

### 91.4 phasewrap

wrap the phase to +/- pi

### 91.5 test\_cwt\_man

### 91.6 test\_phasewrap

### 91.7 test\_wavelet

### 91.8 test\_wavelet2

## **91.9 test\_wavelet\_analysis**

## **91.10 test\_wavelet\_reconstruct**

## **91.11 test\_wtc**

## **91.12 wavelet**

wavelet windows

## **91.13 wavelet\_reconstruct**

iverses wavelet transform for single frequency  
(reconstruction of time series)  
n : window lengths in multiples of filter period  $1/f_0$

## **91.14 wavelet\_transform**

wavelet transform for single frequency  
n : window lengths in multiples of filter period  $1/f_0$

# **92 mathematics**

mathematical functions of various kind

## **92.1 wrapphase**