Manual for Package: mathematics
Revision 2:9M

Karl Kästner

March 28, 2020

Contents

1 calendar **34**
- 1.1 days_per_month 34
- 1.2 isnight 34

2 mathematics **35**
- 2.1 cast_byte_to_integer 35

3 complex-analysis **35**
- 3.1 complex_exp_product_im_im 35
- 3.2 complex_exp_product_im_re 35
- 3.3 complex_exp_product_re_im 35
- 3.4 complex_exp_product_re_re 36
- 3.5 croots 36
- 3.6 root_complex 37
- 3.7 test_imroots 37

4 derivation **37**
- 4.1 derive_acfar1 37
- 4.2 derive_ar2param 37
- 4.3 derive_arc_length 37
- 4.4 derive_fourier_power 37
- 4.5 derive_fourier_power_exp 37
- 4.6 derive_laplacian_curvilinear 37
- 4.7 derive_laplacian_fourier_piecewise_linear 37
- 4.8 derive_logtripdf 38
- 4.9 derive_smooth1d_parametric 38

5 derivation/master **38**
- 5.1 derive_bc_one_sided 38
- 5.2 derive_convergence 38

5.3 derive_error_fdm . . . 38
5.4 derive_fdm_poly . . . 38
5.5 derive_fdm_power . . . 38
5.6 derive_fdm_taylor . . . 38
5.7 derive_fdm_vargrid . . . 38
5.8 derive_fem_2d_mass . . . 39
5.9 derive_fem_error_2d . . . 39
5.10 derive_fem_error_3d . . . 39
5.11 derive_fem_sym_2d . . . 39
5.12 derive_grid_constants . . . 39
5.13 derive_interpolation . . . 39
5.14 derive_laplacian . . . 39
5.15 derive_limit . . . 39
5.16 derive_nc_1d . . . 39
5.17 derive_nc_1d_ . . . 39
5.18 derive_nc_2d . . . 40
5.19 derive_nonuniform_symmetric . . . 40
5.20 derive_richardson . . . 40
5.21 derive_sum . . . 40
5.22 nn . . . 40
5.23 test_derive . . . 40
5.24 test_derive_fdm_poly . . . 40
5.25 test_filter . . . 40
5.26 test_vargrid . . . 40

6 derivation **41**
6.1 simplify_atan . . . 41

7 mathematics **41**
7.1 exp10 . . . 41

8 finance **41**
8.1 derive_skewrnd_walsh_paramter . . . 41
8.2 gbm_cdf . . . 41
8.3 gbm_fit . . . 41
8.4 gbm_fit_old . . . 41
8.5 gbm_inv . . . 41
8.6 gbm_mean . . . 42
8.7 gbm_median . . . 42
8.8 gbm_pdf . . . 42
8.9 gbm_simulate . . . 42
8.10 gbm_skewness . . . 42
8.11 gbm_std . . . 42
8.12 gbm_transform_time_step . . . 42

8.13 put_price_black_scholes 42
8.14 skewgbm_simulate 42
8.15 skewrnd_walsh 42

9 finance/test **43**
9.1 test_gbm 43
9.2 test_gbm_pdf 43
9.3 test_skewrnd_walsh 43

10 fourier/@STFT **43**
10.1 STFT 43
10.2 itransform 43
10.3 stft_ 43
10.4 stftmat 43
10.5 transform 44

11 fourier **44**
11.1 amplitude_from_peak 44
11.2 dftmtx_man 44
11.3 example_fourier_window 44
11.4 fft_derivative 45
11.5 fft_man 45
11.6 fftsmooth 45
11.7 fix_fourier 46
11.8 fourier_axis 46
11.9 fourier_cesaro_correction 46
11.10 fourier_coefficient_piecewise_linear 46
11.11 fourier_coefficient_piecewise_linear_1 47
11.12 fourier_coefficient_ramp3 47
11.13 fourier_coefficient_ramp_pulse 47
11.14 fourier_coefficient_ramp_step 47
11.15 fourier_coefficient_square_pulse 47
11.16 fourier_cubic_interaction_coefficients 47
11.17 fourier_derivative 48
11.18 fourier_expand 48
11.19 fourier_fit 48
11.20 fourier_interpolate 48
11.21 fourier_matrix 48
11.22 fourier_matrix2 48
11.23 fourier_matrix3 48
11.24 fourier_matrix_exp 49
11.25 fourier_multiplicative_interaction_coefficients 49
11.26 fourier_power 49
11.27 fourier_power_exp 49

11.28 fourier_predict . . . 49
11.29 fourier_quadratic_interaction_coefficients . . . 50
11.30 fourier_range . . . 50
11.31 fourier_regress . . . 50
11.32 fourier_resampled_fit . . . 50
11.33 fourier_resampled_predict . . . 50
11.34 fourier_signed_square . . . 50
11.35 fourier_transform . . . 51
11.36 hyperbolic_fourier_box . . . 51
11.37 idftmtx_man . . . 51
11.38 laplace_2d_pwlinear . . . 51
11.39 nanfft . . . 52
11.40 peaks . . . 52
11.41 roots_fourier . . . 52
11.42 spectral_density . . . 52
11.43 test_complex_exp_product . . . 52
11.44 test_fourier_filter . . . 53
11.45 test_idftmtx . . . 53

12 geometry/@Geometry 53

12.1 Geometry . . . 53
12.2 arclength . . . 53
12.3 arclength_old . . . 53
12.4 arclength_old2 . . . 53
12.5 base_point . . . 53
12.6 base_point_limited . . . 54
12.7 centroid . . . 54
12.8 cosa_min_max . . . 54
12.9 cross2 . . . 54
12.10 curvature . . . 54
12.11 ddot . . . 54
12.12 distance . . . 54
12.13 distance2 . . . 54
12.14 dot . . . 55
12.15 edge_length . . . 55
12.16 enclosed_angle . . . 55
12.17 enclosing_triangle . . . 55
12.18 hexagon . . . 55
12.19 inPolygon . . . 55
12.20 inTetra . . . 55
12.21 inTetra2 . . . 55
12.22 inTriangle . . . 56
12.23 intersect . . . 56
12.24 lineintersect . . . 56

12.25 lineintersect1 56
12.26 minimum_distance_lines 56
12.27 mittenpunkt 56
12.28 nagelpoint 56
12.29 onLine 56
12.30 orthocentre 57
12.31 plumb_line 57
12.32 poly_area 57
12.33 poly_edges 57
12.34 poly_set 57
12.35 poly_width 57
12.36 polyxpoly 57
12.37 project_to_curve 58
12.38 quad_isconvex 58
12.39 random_disk 58
12.40 random_simplex 58
12.41 sphere_volume 58
12.42 tetra_volume 58
12.43 tobarycentric 58
12.44 tobarycentric1 58
12.45 tobarycentric2 59
12.46 tobarycentric3 59
12.47 tri_angle 59
12.48 tri_area 59
12.49 tri_centroid 59
12.50 tri_distance_opposit_midpoint 59
12.51 tri_edge_length 59
12.52 tri_edge_midpoint 59
12.53 tri_excircle 60
12.54 tri_height 60
12.55 tri_incircle 60
12.56 tri_isacute 60
12.57 tri_isobtuse 60
12.58 tri_semiperimeter 60
12.59 tri_side_length 60

13 geometry **61**
13.1 Polygon 61
13.2 bounding_box 61
13.3 curvature_1d 61
13.4 cvt 61
13.5 deg_to_frac 61
13.6 ellipse 61
13.7 ellipseX 62

13.8 ellipseY 62
13.9 first_intersect 62
13.10 golden_ratio 62
13.11 hypot3 62
13.12 meanangle 62
13.13 meanangle2 62
13.14 meanangle3 62
13.15 meanangle4 63
13.16 mediananangle 63
13.17 medianangle2 63
13.18 pilim 63
13.19 streamline_radius_of_curvature 63

14 histogram/@Histogram 63

14.1 2x 63
14.2 Histogram 64
14.3 bimodes 64
14.4 cdf 64
14.5 cdfS 64
14.6 chi2test 64
14.7 cmoment 64
14.8 cmomentS 64
14.9 entropy 64
14.10 entropyS 64
14.11 iquantile 64
14.12 kstest 65
14.13 kurtosis 65
14.14 kurtosisS 65
14.15 mean 65
14.16 meanS 65
14.17 median 65
14.18 medianS 65
14.19 mode 65
14.20 modeS 65
14.21 moment 65
14.22 momentS 66
14.23 pdf 66
14.24 quantile 66
14.25 quantileS 66
14.26 setup 66
14.27 skewness 66
14.28 skewnessS 66
14.29 stairs 66
14.30 stairsS 66

14.31 std . . . 66
14.32 stdS . . . 67
14.33 var . . . 67
14.34 varS . . . 67

15 histogram **67**
15.1 hist_man . . . 67
15.2 histadapt . . . 67
15.3 histconst . . . 67
15.4 pdf_poly . . . 67
15.5 plotcdf . . . 67
15.6 test_histogram . . . 67

16 linear-algebra **68**
16.1 averaging_matrix_2 . . . 68
16.2 colnorm . . . 68
16.3 condest_ . . . 68

17 linear-algebra/coordinate-transformation **68**
17.1 barycentric2cartesian . . . 68
17.2 barycentric2cartesian3 . . . 68
17.3 cartesian2barycentric . . . 68
17.4 cartesian_to_unit_triangle_basis . . . 68
17.5 ellipsoid2geoid . . . 68
17.6 example_approximate_utm_conversion . . . 69
17.7 latlon2utm . . . 69
17.8 latlon2utm_simple . . . 69
17.9 lowrance_mercator_to_wgs84 . . . 69
17.10 nmea2utm . . . 69
17.11 sn2xy . . . 69
17.12 unit_triangle_to_cartesian . . . 69
17.13 utm2latlon . . . 69
17.14 xy2nt . . . 70
17.15 xy2sn . . . 70
17.16 xy2sn_java . . . 70
17.17 xy2sn_old . . . 70

18 linear-algebra **70**
18.1 det2x2 . . . 70
18.2 det3x3 . . . 70
18.3 det4x4 . . . 71
18.4 diag2x2 . . . 71
18.5 eig2x2 . . . 71

19 linear-algebra/eigenvalue **71**
19.1 eig_bisection . . . 71
19.2 eig_inverse . . . 71
19.3 eig_inverse_iteration . . . 71
19.4 eig_power_iteration . . . 71

20 linear-algebra/eigenvalue/jacobi-davidson **71**
20.1 afun_jdm . . . 71
20.2 davidson . . . 72
20.3 jacobi_davidson . . . 72
20.4 jacobi_davidson_qr . . . 72
20.5 jacobi_davidson_qz . . . 72
20.6 jacobi_davidson_simple . . . 72
20.7 jdqr . . . 72
20.8 jdqr_sleijpen . . . 76
20.9 jdqr_vorst . . . 79
20.10 jdqz . . . 82
20.11 mfunc_jdm . . . 86
20.12 mgs . . . 86
20.13 minres_ . . . 87
20.14 mv_jacobi_davidson . . . 87

21 linear-algebra **87**
21.1 first . . . 87
21.2 gershgorin_circle . . . 87
21.3 haussdorff . . . 87
21.4 ieig2x2 . . . 87
21.5 inv2x2 . . . 87
21.6 inv3x3 . . . 88
21.7 inv4x4 . . . 88

22 linear-algebra/lanczos **88**
22.1 arnoldi . . . 88
22.2 arnoldi_new . . . 88
22.3 eigs_lanczos_man . . . 88
22.4 lanczos . . . 88
22.5 lanczos_ . . . 88
22.6 lanczos_biorthogonal . . . 88
22.7 lanczos_biorthogonal_improved . . . 88
22.8 lanczos_ghep . . . 89
22.9 mv_lanczos . . . 89
22.10 reorthogonalise . . . 89
22.11 test_lanczos . . . 89

23 linear-algebra/linear-systems **89**
23.1 gmres_man 89
23.2 minres_recycle 89

24 linear-algebra **89**
24.1 lpmean 89
24.2 lpnorm 89
24.3 matvec3 90
24.4 max2d 90
24.5 mpoweri 90
24.6 mtimes2x2 90
24.7 mtimes3x3 90
24.8 nannorm 90
24.9 nanshift 90
24.10 nl 90
24.11 normalise 91
24.12 normalize1 91
24.13 normrows 91
24.14 orth2 91
24.15 orth_man 91
24.16 orthogonalise 91
24.17 paddext 91
24.18 paddval1 92
24.19 paddval2 92

25 linear-algebra/polynomial **92**
25.1 chebychev 92
25.2 piecewise_polynomial 92
25.3 roots1 92
25.4 roots2 92
25.5 roots2poly 92
25.6 roots3 92
25.7 roots4 93
25.8 test_roots4 93
25.9 vanderi_1d 93

26 linear-algebra **93**
26.1 randrot 93
26.2 right 93
26.3 rot2 93
26.4 rot2dir 93
26.5 rot3 93
26.6 rotR 94
26.7 rownorm 94

26.8 simmilarity_matrix 94
26.9 spnorm 94
26.10 spzeros 94
26.11 test_roots3 94
26.12 transform_minmax 94
26.13 transpose3 94
26.14 transposeall 94

27 logic **95**
27.1 bitor_man 95

28 master/plot **95**
28.1 attach_boundary_value 95
28.2 cartesian_polar 95
28.3 img_vargrid 95
28.4 plot_basis_functions 95
28.5 plot_convergence 95
28.6 plot_dof 95
28.7 plot_eigenbar 95
28.8 plot_error_estimation 96
28.9 plot_error_estimation_2 96
28.10 plot_error_fem 96
28.11 plot_fdm_kernel 96
28.12 plot_fdm_vs_fem 96
28.13 plot_fem_accuracy 96
28.14 plot_function_and_grid 96
28.15 plot_hat 96
28.16 plot_hydrogen_wf 96
28.17 plot_mesh 96
28.18 plot_mesh_2 97
28.19 plot_refine 97
28.20 plot_refine_3d 97
28.21 plot_runtime 97
28.22 plot_spectrum 97
28.23 plot_wavefunction 97

29 master/ported **97**
29.1 assemble_2d_dphi_dphi 97
29.2 assemble_2d_phi_phi 97
29.3 assemble_3d_dphi_dphi 97
29.4 assemble_3d_phi_phi 98
29.5 dV_2d_ 98
29.6 derivative_2d 98
29.7 derivative_3d 98

29.8 element_neighbour_2d 98
29.9 prefetch_2d_ 98
29.10 promote_2d_3_10 98
29.11 promote_2d_3_15 98
29.12 promote_2d_3_21 98
29.13 promote_2d_3_6 98
29.14 promote_3d_4_10 99
29.15 promote_3d_4_20 99
29.16 promote_3d_4_35 99
29.17 vander_2d 99
29.18 vander_3d 99

30 master/sandbox **99**
30.1 adapt 99
30.2 assoc_laguerre 99
30.3 assoc_legendre 99
30.4 c23 99

31 master/sandbox/cg **100**
31.1 cg 100
31.2 cg_coef_to_poly 100
31.3 errmat 100
31.4 lanczos 100
31.5 laplacian_2d 100
31.6 test_cg_eigs 100
31.7 test_lanczos 100

32 master/sandbox **100**
32.1 condition_number_higher_order 100
32.2 confinement_dat 100
32.3 convergence_2d_3d 101
32.4 convergence_matrix_powers 101
32.5 cut_out 101
32.6 derivative_2d 101
32.7 derivative_3d 101
32.8 dummy 101
32.9 eig_error 101
32.10 eigs_fix 101
32.11 energy_level 101
32.12 equalise 101
32.13 example_int64 102

33 master/sandbox/fem-matlab **102**
33.1 boundary_circle 102

33.2 boundary_rectangle . . . 102
33.3 geometry_circle_with_hole . . . 102
33.4 geometry_rectangle . . . 102

34 master/sandbox **102**
34.1 fem_2d_estimate_error . . . 102
34.2 fem_assemble_scratch . . . 102
34.3 fem_s . . . 103
34.4 fourier_h . . . 103
34.5 grad_2d . . . 103
34.6 grad_3d . . . 103
34.7 gradient . . . 103
34.8 harmonic_oscillator . . . 103
34.9 hydrogen_2d_analytic . . . 103
34.10 hydrogen_boxed . . . 103
34.11 hydrogen_boxed_old . . . 103
34.12 hydrogen_wave . . . 103
34.13 hydrogen_wf . . . 104
34.14 ichol_man . . . 104
34.15 known_eigenvalue . . . 104
34.16 kron_man . . . 104
34.17 laguerre . . . 104
34.18 laplacian_arbitrary_order_old . . . 104
34.19 laplacian_convergence . . . 104
34.20 laplacian_cut_out . . . 104
34.21 laplacian_cylindrical . . . 104
34.22 laplacian_non_uniform_old . . . 104
34.23 laplacian_polar . . . 105
34.24 laplacian_simple . . . 105
34.25 lderivative_3d . . . 105
34.26 list_dat . . . 105
34.27 matlab-horner . . . 105
34.28 mesh_to_grid_2d_3 . . . 105
34.29 mg_mat . . . 105
34.30 mv . . . 105
34.31 orth2 . . . 105
34.32 partial_derivative_2d . . . 105
34.33 partition_function . . . 106
34.34 partition_function_old . . . 106
34.35 poisson . . . 106
34.36 poisson_fem . . . 106
34.37 potential . . . 106
34.38 powerc . . . 106
34.39 quick_newihbour . . . 106

34.40 radial . . . 106
34.41 radial_convergence . . . 106
34.42 radial_wafefunction . . . 106
34.43 refine_2d . . . 107
34.44 refine_3d . . . 107
34.45 relerr . . . 107
34.46 restore_cw . . . 107
34.47 runtime_bm . . . 107
34.48 rydberg . . . 107
34.49 s_old . . . 107
34.50 snorm . . . 107
34.51 spherical_harmonic . . . 107
34.52 split_eig . . . 107
34.53 sum1 . . . 108
34.54 sum3 . . . 108

35 master/sandbox/summation 108
35.1 acc . . . 108
35.2 add . . . 108
35.3 ape . . . 108
35.4 mmul_accurately . . . 108
35.5 sum_kahan . . . 108
35.6 sum_pairwise . . . 108
35.7 test_sum . . . 108

36 master/sandbox 109
36.1 test_convergence_ill_conditioned . . . 109
36.2 test_fem_1d . . . 109
36.3 test_fem_2d . . . 109
36.4 test_fem_3d . . . 109
36.5 test_increase . . . 109
36.6 test_lanczos_shift . . . 109
36.7 test_ldl . . . 109
36.8 test_power . . . 109
36.9 trefethen_p8_fdm . . . 109
36.10 wavefunc . . . 110
36.11 xgrid . . . 110

37 number-theory 110
37.1 ceiln . . . 110
37.2 digitsb . . . 110
37.3 floorn . . . 110
37.4 iseven . . . 110
37.5 multichoosek . . . 110

37.6 nchoosek_man . . . 111
37.7 pythagorean_triple . . . 111
37.8 roundn . . . 111

38 numerical-methods/differentiation 111
38.1 derivative1 . . . 111
38.2 derivative2 . . . 111

39 numerical-methods/finite-difference 111
39.1 cdiff . . . 111
39.2 cdiffb . . . 112
39.3 cmean . . . 112
39.4 derivative_matrix_1_1d . . . 112
39.5 derivative_matrix_2_1d . . . 112
39.6 derivative_matrix_2d . . . 112
39.7 derivative_matrix_curvilinear . . . 112
39.8 derivative_matrix_curvilinear_2 . . . 112
39.9 difference_kernel . . . 113
39.10 distmat . . . 113
39.11 gradpde2d . . . 113
39.12 laplacian . . . 113
39.13 laplacian_fdm . . . 113
39.14 left . . . 113
39.15 lrmean . . . 113

40 numerical-methods/finite-difference/master 114
40.1 fdm_adaptive_grid . . . 114
40.2 fdm_adaptive_refinement_old . . . 114
40.3 fdm_assemble_d1_2d . . . 114
40.4 fdm_assemble_d2_2d . . . 114
40.5 fdm_confinement . . . 114
40.6 fdm_d_vargrid . . . 114
40.7 fdm_h_unstructured . . . 114
40.8 fdm_hydrogen_vargrid . . . 114
40.9 fdm_mark_unstructured_2d . . . 114
40.10 fdm_plot . . . 115
40.11 fdm_plot_series . . . 115
40.12 fdm_refine_2d . . . 115
40.13 fdm_refine_3d . . . 115
40.14 fdm_refine_unstructured_2d . . . 115
40.15 fdm_schroedinger_2d . . . 115
40.16 fdm_schroedinger_3d . . . 115
40.17 relocate . . . 115

41 numerical-methods/finite-difference **115**
41.1 mid 115
41.2 pwmid 116
41.3 ratio 116
41.4 steplength 116
41.5 swapoddeven 116
41.6 test_derivative_matrix_2d 116
41.7 test_derivative_matrix_curvilinear 116
41.8 test_difference_kernel 116

42 numerical-methods/finite-element **116**
42.1 Mesh_2d_java 116
42.2 Tree_2d_java 117
42.3 assemble_1d_dphi_dphi 117
42.4 assemble_1d_phi_phi 117
42.5 assemble_2d_dphi_dphi_java 117
42.6 assemble_2d_phi_phi_java 117
42.7 assemble_3d_dphi_dphi_java 117
42.8 assemble_3d_phi_phi_java 117
42.9 boundary_1d 117
42.10 boundary_2d 117
42.11 boundary_3d 117
42.12 check_area_2d 118
42.13 circmesh 118
42.14 cropradius 118
42.15 display_2d 118
42.16 display_3d 118
42.17 distort 118
42.18 err_2d 118
42.19 estimate_err_2d_3 118
42.20 example_1d 118
42.21 example_2d 118
42.22 explode 119
42.23 fem_2d 119
42.24 fem_2d_heuristic_mesh 119
42.25 fem_get_2d_radial 119
42.26 fem_interpolation 119
42.27 fem_plot_1d 119
42.28 fem_plot_1d_series 119
42.29 fem_plot_2d 119
42.30 fem_plot_2d_series 119
42.31 fem_plot_3d 119
42.32 fem_plot_3d_series 120
42.33 fem_plot_confine_series 120

42.34 fem_radial . . . 120
42.35 flip_2d . . . 120
42.36 get_mesh_arrays . . . 120
42.37 hashkey . . . 120

43 numerical-methods/finite-element/int **120**
43.1 int_1d_gauss . . . 120
43.2 int_1d_gauss_1 . . . 120
43.3 int_1d_gauss_2 . . . 120
43.4 int_1d_gauss_3 . . . 121
43.5 int_1d_gauss_4 . . . 121
43.6 int_1d_gauss_5 . . . 121
43.7 int_1d_gauss_6 . . . 121
43.8 int_1d_gauss_lobatto . . . 121
43.9 int_1d_nc_2 . . . 121
43.10 int_1d_nc_3 . . . 121
43.11 int_1d_nc_4 . . . 121
43.12 int_1d_nc_5 . . . 121
43.13 int_1d_nc_6 . . . 121
43.14 int_1d_nc_7 . . . 122
43.15 int_1d_nc_7_hardy . . . 122
43.16 int_2d_gauss_1 . . . 122
43.17 int_2d_gauss_12 . . . 122
43.18 int_2d_gauss_13 . . . 122
43.19 int_2d_gauss_16 . . . 122
43.20 int_2d_gauss_25 . . . 122
43.21 int_2d_gauss_3 . . . 122
43.22 int_2d_gauss_33 . . . 122
43.23 int_2d_gauss_6 . . . 122
43.24 int_2d_gauss_7 . . . 123
43.25 int_2d_gauss_9 . . . 123
43.26 int_2d_nc_10 . . . 123
43.27 int_2d_nc_15 . . . 123
43.28 int_2d_nc_21 . . . 123
43.29 int_2d_nc_3 . . . 123
43.30 int_2d_nc_6 . . . 123
43.31 int_3d_gauss_1 . . . 123
43.32 int_3d_gauss_11 . . . 123
43.33 int_3d_gauss_14 . . . 123
43.34 int_3d_gauss_15 . . . 124
43.35 int_3d_gauss_24 . . . 124
43.36 int_3d_gauss_4 . . . 124
43.37 int_3d_gauss_45 . . . 124
43.38 int_3d_gauss_5 . . . 124

43.39 int_3d_nc_11 124
43.40 int_3d_nc_4 124
43.41 int_3d_nc_6 124
43.42 int_3d_nc_8 124

44 numerical-methods/finite-element 125
44.1 interpolation_matrix 125
44.2 mark 125
44.3 mark_1d 125
44.4 mesh_1d_uniform 125
44.5 mesh_3d_uniform 125
44.6 mesh_interpolate 125
44.7 neighbour_1d 125
44.8 old 125
44.9 pdeeig_1d 125
44.10 pdeeig_2d 126
44.11 pdeeig_3d 126
44.12 polynomial_derivative_1d 126
44.13 potential_const 126
44.14 potential_coulomb 126
44.15 potential_harmonic_oscillator 126
44.16 project_circle 126
44.17 project_rectangle 126
44.18 promote_1d_2_3 126
44.19 promote_1d_2_4 126
44.20 promote_1d_2_5 127
44.21 promote_1d_2_6 127
44.22 quadrilaterate 127
44.23 recalculate_regularity_2d 127
44.24 refine_1d 127
44.25 refine_2d_21 127
44.26 refine_2d_structural 127
44.27 regularity_1d 127
44.28 regularity_2d 127
44.29 regularity_3d 128
44.30 relocate_2d 128
44.31 test_circmesh 128
44.32 test_hermite 128
44.33 tri_assign_points 128
44.34 triangulation_uniform 128
44.35 vander_1d 128
44.36 vanderd_1d 128
44.37 vanderi_1d 128

45 numerical-methods/finite-volume/@Advection 129
45.1 Advection . . . 129
45.2 dot_advection . . . 129

46 numerical-methods/finite-volume/@Burgers 129
46.1 burgers_split . . . 129
46.2 dot_burgers_fdm . . . 129
46.3 dot_burgers_fft . . . 129

47 numerical-methods/finite-volume/@Finite_Volume 129
47.1 Finite_Volume . . . 129
47.2 apply_bc . . . 130
47.3 solve . . . 130
47.4 step_split_strang . . . 130
47.5 step_unsplit . . . 130

48 numerical-methods/finite-volume/@Flux_Limiter 130
48.1 Flux_Limiter . . . 130
48.2 beam_warming . . . 130
48.3 fromm . . . 131
48.4 lax_wendroff . . . 131
48.5 minmod . . . 131
48.6 monotized_central . . . 131
48.7 muscl . . . 131
48.8 superbee . . . 131
48.9 upwind . . . 131
48.10 vanLeer . . . 132

49 numerical-methods/finite-volume/@KDV 132
49.1 dot_kdv_fdm . . . 132
49.2 dot_kdv_fft . . . 132
49.3 kdv_split . . . 132

50 numerical-methods/finite-volume/@Reconstruct_Average_Evolve 132
50.1 Reconstruct_Average_Evolve . . . 132
50.2 advect_highres . . . 133
50.3 advect_lowress . . . 133

51 numerical-methods/finite-volume 133
51.1 Godunov . . . 133
51.2 Lax_Friedrich . . . 133
51.3 Measure . . . 133
51.4 Roe . . . 133
51.5 fv_swe . . . 134
51.6 staggered_euler . . . 134

51.7 staggered_grid 134

52 numerical-methods **134**

52.1 grid2quad 134

53 numerical-methods/integration **134**

53.1 cumintL 134

53.2 cumintR 134

53.3 int_trapezoidal 134

54 numerical-methods/interpolation/@Kriging **135**

54.1 Kriging 135

54.2 estimate_semivariance 135

54.3 interpolate_ 135

55 numerical-methods/interpolation/@RegularizedInterpolator1 **135**

55.1 RegularizedInterpolator1 135

55.2 init 135

56 numerical-methods/interpolation/@RegularizedInterpolator2 **136**

56.1 RegularizedInterpolator2 136

56.2 init 136

57 numerical-methods/interpolation/@RegularizedInterpolator3 **136**

57.1 RegularizedInterpolator3 136

57.2 init 136

58 numerical-methods/interpolation **136**

58.1 IDW 136

58.2 IPoly 136

58.3 IRBM 137

58.4 ISparse 137

58.5 Inn 137

58.6 Interpolator 137

58.7 fixnan 137

58.8 idw1 137

58.9 idw2 137

58.10 inner2outer 138

58.11 inner2outer2 138

58.12 interp1_limited 138

58.13 interp1_man 138

58.14 interp1_save 138

58.15 interp1_slope 138

58.16 interp1_smooth 139

58.17 interp1_unique 139

58.18 interp2_man 139
58.19 interp_angle 139
58.20 interp_fourier 139
58.21 interp_fourier_batch 139
58.22 interp_sn 139
58.23 interp_sn2 140
58.24 interp_sn3 140
58.25 interp_sn_ 140
58.26 limit_by_distance_1d 140
58.27 resample1 140
58.28 resample_d_min 140
58.29 resample_vector 140
58.30 test_interp1_limited 140

59 numerical-methods **141**
59.1 inverse_complex 141

60 numerical-methods/ode **141**
60.1 bvp2_check_arguments 141
60.2 bvp2c 141
60.3 bvp2c2 141
60.4 bvp2fdm 142
60.5 bvp2wavetrain 142
60.6 bvp2wavetwopass 142
60.7 ivp_euler_forward 142
60.8 ivprk2 143
60.9 ode2_matrix 143
60.10 ode2characteristic 143
60.11 step_trapezoidal 143
60.12 test_bvp2 143

61 numerical-methods/optimisation **143**
61.1 armijo_stopping_criterion 143
61.2 astar 143
61.3 binsearch 144
61.4 bisection 144
61.5 box1 144
61.6 box2 144
61.7 cauchy 144
61.8 cauchy2 144
61.9 directional_derivative 144
61.10 dud 145
61.11 extreme3 145
61.12 extreme_quadratic 145

61.13 ftest 145
61.14 fzero_bisect 145
61.15 fzero_newton 145
61.16 grad 146
61.17 hessian 146
61.18 hessian_from_gradient 146
61.19 hessian_projected 146
61.20 line_search 146
61.21 line_search2 146
61.22 line_search_polynomial 147
61.23 line_search_polynomial2 147
61.24 line_search_quadratic 147
61.25 line_search_quadratic2 147
61.26 line_search_wolfe 148
61.27 ls_bgfs 148
61.28 ls_broyden 148
61.29 ls_generalized_secant 148
61.30 nlcg 148
61.31 nlls 149
61.32 picard 149
61.33 poly_extrema 149
61.34 quadratic_function 149
61.35 quadratic_programming 149
61.36 quadratic_step 149
61.37 rosenbrock 149
61.38 sqrt_heron 150
61.39 test_directional_derivative 150
61.40 test_dud 150
61.41 test_fzero_newton 150
61.42 test_line_search_quadratic2 150
61.43 test_ls_generalized_secant 150
61.44 test_nlcg_6_order 150
61.45 test_nlls 150

62 numerical-methods/pde 150
62.1 laplacian2d_fundamental_solution 150

63 numerical-methods/piecewise-polynomials 151
63.1 Hermite1 151
63.2 hp2_fit 151
63.3 hp2_predict 151
63.4 hp_predict 151
63.5 hp_regress 151
63.6 lp_count 151

63.7 lp_predict . . . 152
63.8 lp_regress . . . 152
63.9 lp_regress_ . . . 152

64 regression/@PolyOLS **152**
64.1 PolyOLS . . . 152
64.2 coefftest . . . 152
64.3 detrend . . . 152
64.4 fit . . . 152
64.5 fit_ . . . 152
64.6 predict . . . 153
64.7 predict_ . . . 153
64.8 slope . . . 153

65 regression/@PowerLS **153**
65.1 PowerLS . . . 153
65.2 fit . . . 153
65.3 predict . . . 153
65.4 predict_ . . . 153

66 regression/@Theil **154**
66.1 Theil . . . 154
66.2 detrend . . . 154
66.3 fit . . . 154
66.4 predict . . . 154
66.5 slope . . . 154

67 regression **154**
67.1 Theil_Multivariate . . . 154
67.2 areg . . . 155
67.3 ginireg . . . 155
67.4 hesssimplereg . . . 155
67.5 l1lin . . . 155
67.6 lsq_sparam . . . 155
67.7 polyfitd . . . 155
67.8 regression_method_of_moments . . . 156
67.9 robustlinreg . . . 156
67.10 theil2 . . . 156
67.11 theil_generalised . . . 156
67.12 total_least_squares . . . 156
67.13 weighted_median_regression . . . 156

68 set-theory **157**
68.1 issubset . . . 157

69 signal-processing **157**
69.1 acf_effective_sample_size . . . 157
69.2 acf_genton . . . 157
69.3 acfar1 . . . 157
69.4 acfar1_2 . . . 157
69.5 acfar2 . . . 157
69.6 acfar2_2 . . . 158
69.7 ar1_cutoff_frequency . . . 158
69.8 ar1_effective_sample_size . . . 158
69.9 ar1_mse_mu_single_sample . . . 158
69.10 ar1_mse_pop . . . 158
69.11 ar1_mse_range . . . 158
69.12 ar1_spectrum . . . 158
69.13 ar1_to_tikhonov . . . 158
69.14 ar1_var_factor . . . 159
69.15 ar1_var_factor_ . . . 159
69.16 ar1_var_range2 . . . 159
69.17 ar1delay . . . 159
69.18 ar1delay_old . . . 159
69.19 ar2conv . . . 159
69.20 ar2dof . . . 160
69.21 ar2param . . . 160
69.22 asymwin . . . 160
69.23 autocorr_fft . . . 160
69.24 bandpass . . . 160
69.25 bandpass2 . . . 160
69.26 bartlett . . . 160
69.27 bartlett_spectrogram . . . 161
69.28 bin1d . . . 161
69.29 bin2d . . . 161
69.30 binormrnd . . . 161
69.31 conv1_man . . . 161
69.32 conv2_man . . . 161
69.33 conv2z . . . 161
69.34 conv30 . . . 162
69.35 conv_ . . . 162
69.36 conv_centered . . . 162
69.37 convz . . . 162
69.38 cosexpdelay . . . 162
69.39 csmooth . . . 162
69.40 daniell_window . . . 162
69.41 danielle_window . . . 163
69.42 db2neper . . . 163
69.43 db2power . . . 163

69.44 derive_danielle_weight . . . 163
69.45 derive_limit_0_acfar . . . 163
69.46 detect_peak . . . 163
69.47 digital_low_pass_filter . . . 163
69.48 doublesum_ij . . . 164
69.49 effective_sample_size_to_ar1 . . . 164
69.50 filt_hodges_lehman . . . 164
69.51 filter1 . . . 164
69.52 filter2 . . . 164
69.53 filter_ . . . 164
69.54 filteriir . . . 164
69.55 filterp . . . 165
69.56 filterp1 . . . 165
69.57 filterstd . . . 165
69.58 firls_man . . . 165
69.59 flattopwin . . . 166
69.60 frequency_response_boxcar . . . 166
69.61 freqz_boxcar . . . 166
69.62 gaussfilt1 . . . 166
69.63 hanchangewin . . . 166
69.64 hanchangewin2 . . . 166
69.65 hanwin . . . 166
69.66 hanwin_ . . . 166
69.67 highpass . . . 167
69.68 kaiserwin . . . 167
69.69 kalman . . . 167
69.70 lanczoswin . . . 167
69.71 last . . . 167
69.72 lowpass . . . 167
69.73 lowpass2 . . . 167
69.74 lowpass_iir . . . 167
69.75 lowpass_iir_symmetric . . . 168
69.76 lowpassfilter2 . . . 168
69.77 maxfilt1 . . . 168
69.78 meanfilt1 . . . 168
69.79 medfilt1_man . . . 168
69.80 medfilt1_man2 . . . 168
69.81 medfilt1_padded . . . 168
69.82 medfilt1_reduced . . . 168
69.83 mid_term_single_sample . . . 169
69.84 minfilt1 . . . 169
69.85 mu2ar1 . . . 169
69.86 mysmooth . . . 169
69.87 nanautocorr . . . 169

69.88 nanmedfilt1 169
69.89 neper2db 169
69.90 peaks_man 169
69.91 polyfilt1 170
69.92 qmedfilt1 170
69.93 randar1 170
69.94 randar1_dual 170
69.95 randar2 170
69.96 randarp 170
69.97 range_window 170
69.98 rectwin 170
69.99 recursive_sum 171
69.100 select_range 171
69.101 smooth1d_parametric 171
69.102 smooth2 171
69.103 smooth_man 171
69.104 smooth_parametric 171
69.105 smooth_parametric2 171
69.106 smooth_with_splines 171
69.107 smoothfft 172
69.108 spectrogram 172
69.109 std_window 172
69.110 sum_i_lag 172
69.111 sum_ii 172
69.112 sum_ii_ 172
69.113 sum_ij 172
69.114 sum_ij_ 172
69.115 sum_ij_partial_ 173
69.116 sum_multivar 173
69.117 test_acfar1 173
69.118 test_acfar1_2 173
69.119 test_acfar1_3 173
69.120 test_acfar1_4 173
69.121 test_acfar2 173
69.122 test_ar1_var_factor 173
69.123 test_ar1_var_factor_2 173
69.124 test_ar1_var_mu_single_sample 173
69.125 test_ar1_var_pop 174
69.126 test_ar1_var_pop_1 174
69.127 test_ar1delay 174
69.128 test_bivariate_covariance_term 174
69.129 test_convexity 174
69.130 test_lanczoswin 174
69.131 test_madcorr 174

69.132 test_randar1 . . . 174
69.133 test_randar1_multivariate . . . 174
69.134 test_randar2 . . . 174
69.135 test_sum_ij . . . 175
69.136 test_sum_multivar . . . 175
69.137 test_trifilt1 . . . 175
69.138 test_wautocorr . . . 175
69.139 test_wavelet_transform . . . 175
69.140 test_wordfilt . . . 175
69.141 test_xar1_mid_term . . . 175
69.142 tikhonov_to_ar1 . . . 175
69.143 trapwin . . . 175
69.144 trifilt1 . . . 176
69.145 triwin . . . 176
69.146 triwin2 . . . 176
69.147 varar1 . . . 176
69.148 welch_spectrogram . . . 176
69.149 wfilt . . . 176
69.150 winbandpass . . . 176
69.151 window_make_odd . . . 176
69.152 winfilt0 . . . 177
69.153 winlength . . . 177
69.154 wmeanfilt . . . 177
69.155 wmedfilt . . . 177
69.156 wordfilt . . . 177
69.157 wordfilt_edgeworth . . . 177
69.158 xar1 . . . 177
69.159 xcorr_man . . . 178

70 sorting **178**
70.1 sort2 . . . 178
70.2 sort2d . . . 178

71 special-functions **178**
71.1 bessel_sphere . . . 178
71.2 digamma_man . . . 178
71.3 hankel_sphere . . . 178
71.4 hermite . . . 179
71.5 legendre_man . . . 179
71.6 neumann_sphere . . . 179

72 statistics **179**
72.1 atan_s2 . . . 179
72.2 beta_mode_to_parameter . . . 179

72.3 coefficient_of_determination 179
72.4 conditional_expectation_normal 179
72.5 correlation_confidence_pearson 180

73 statistics/distributions 180
73.1 PDF 180
73.2 binorm_separation_coefficient 180
73.3 binormcdf 180
73.4 binormfit 180
73.5 binormpdf 180
73.6 edgeworth_cdf 180
73.7 edgeworth_pdf 181
73.8 logn_mode2param 181
73.9 logn_param2mode 181
73.10 lognpdf_ 181
73.11 pdfsample 181
73.12 t2cdf 181
73.13 t2inv 181

74 statistics 182
74.1 example_standard_error_of_sample_quantiles 182
74.2 f_var_finite 182
74.3 gamma_mode_to_parameter 182
74.4 gaussfit3 182
74.5 gaussfit_quantile 182
74.6 hodges_lehmann_correlation 182
74.7 hodges_lehmann_dispersion 182

75 statistics/information-theory 183
75.1 akaike_information_criterion 183
75.2 bayesian_information_criterion 183

76 statistics 183
76.1 kurtncdf 183
76.2 kurtnpdf 183
76.3 kurtosis_bias_corrected 183
76.4 limit 184
76.5 logfactorial 184
76.6 loglogpdf 184
76.7 lognfit_quantile 184
76.8 logskewcdf 184
76.9 logskewpdf 184

77 statistics/logu 184

77.1 lambertw_numeric . . . 184
77.2 logtrialtcdf . . . 184
77.3 logtrialtinv . . . 185
77.4 logtrialtmean . . . 185
77.5 logtrialtpdf . . . 185
77.6 logtrialtrnd . . . 185
77.7 logtricdf . . . 185
77.8 logtriinv . . . 185
77.9 logtrimean . . . 186
77.10 logtripdf . . . 186
77.11 logtrirnd . . . 186
77.12 logucdf . . . 186
77.13 logucm . . . 186
77.14 loguinv . . . 186
77.15 logumean . . . 186
77.16 logupdf . . . 186
77.17 logurnd . . . 187
77.18 loguvar . . . 187
77.19 medlogu . . . 187
77.20 test_logurnd . . . 187
77.21 tricdf . . . 187
77.22 triinv . . . 187
77.23 trimedian . . . 187
77.24 tripdf . . . 187
77.25 trirnd . . . 188

78 statistics **188**
78.1 maxnnormals . . . 188
78.2 midrange . . . 188
78.3 minavg . . . 188
78.4 mode_man . . . 188

79 statistics/moment-statistics **188**
79.1 autocorr_man3 . . . 188
79.2 autocorr_man4 . . . 189
79.3 autocorr_man5 . . . 189
79.4 blockserr . . . 189
79.5 comoment . . . 189
79.6 corr_man . . . 189
79.7 cov_man . . . 190
79.8 dof . . . 190
79.9 edgeworth_quantile . . . 190
79.10 effective_sample_size . . . 190
79.11 f_correlation . . . 190

79.12 f_finite . . . 190
79.13 lmean . . . 190
79.14 lmoment . . . 191
79.15 maskmean . . . 191
79.16 masknanmean . . . 191
79.17 mean1 . . . 191
79.18 mean_man . . . 191
79.19 mse . . . 191
79.20 nanautocorr_man1 . . . 191
79.21 nanautocorr_man2 . . . 191
79.22 nanautocorr_man4 . . . 192
79.23 nancorr . . . 192
79.24 nancumsum . . . 192
79.25 nanlmean . . . 192
79.26 nanr2 . . . 192
79.27 nanrms . . . 192
79.28 nanrmse . . . 192
79.29 nanserr . . . 193
79.30 nanwmean . . . 193
79.31 nanwstd . . . 193
79.32 nanwvar . . . 193
79.33 nanxcorr . . . 193
79.34 pearson . . . 193
79.35 pearson_to_kendall . . . 193
79.36 pool_samples . . . 194
79.37 qmean . . . 194
79.38 range_mean . . . 194
79.39 rmse_ . . . 194
79.40 serr . . . 194
79.41 serr1 . . . 194
79.42 test_qskew . . . 194
79.43 test_qstd_qskew_optimal_p . . . 194
79.44 wautocorr . . . 195
79.45 wcorr . . . 195
79.46 wcov . . . 195
79.47 wdof . . . 195
79.48 wkurt . . . 195
79.49 wmean . . . 195
79.50 wrms . . . 196
79.51 wserr . . . 196
79.52 wskew . . . 196
79.53 wstd . . . 196
79.54 wvar . . . 196

80 statistics **196**
80.1 nangeomean . . . 196
80.2 nangeostd . . . 196

81 statistics/nonparametric-statistics **197**
81.1 kernel1d . . . 197
81.2 kernel2d . . . 197

82 statistics **197**
82.1 normmoment . . . 197
82.2 normpdf2 . . . 197

83 statistics/order-statistics **197**
83.1 hodges_lehmann_location . . . 197
83.2 kendall . . . 198
83.3 kendall_to_pearson . . . 198
83.4 mad2sd . . . 198
83.5 madcorr . . . 198
83.6 median2_holder . . . 198
83.7 median_ci . . . 198
83.8 median_man . . . 198
83.9 mediani . . . 199
83.10 nanmadcorr . . . 199
83.11 nanwmedian . . . 199
83.12 nanwquantile . . . 199
83.13 oja_median . . . 199
83.14 qkurtosis . . . 199
83.15 qmoments . . . 200
83.16 qskew . . . 200
83.17 qskewq . . . 200
83.18 qstdq . . . 200
83.19 quantile1_optimisation . . . 200
83.20 quantile2_breckling . . . 200
83.21 quantile2_chaudhuri . . . 200
83.22 quantile2_projected . . . 201
83.23 quantile2_projected2 . . . 201
83.24 quantile_envelope . . . 201
83.25 quantile_regression_simple . . . 201
83.26 ranking . . . 201
83.27 spatial_median . . . 201
83.28 spatial_quantile . . . 201
83.29 spatial_quantile2 . . . 201
83.30 spatial_quantile3 . . . 202
83.31 spatial_rank . . . 202

83.32 spatial_sign . . . 202
83.33 spatial_signed_rank . . . 202
83.34 spearman . . . 202
83.35 spearman_rank . . . 202
83.36 spearman_to_pearson . . . 202
83.37 wmedian . . . 202
83.38 wquantile . . . 203

84 statistics **203**
84.1 qstd . . . 203
84.2 quantile_extrap . . . 203

85 statistics/random-number-generation **203**
85.1 laplacernd . . . 203
85.2 randc . . . 203
85.3 skewness2param . . . 203
85.4 skewpdf_central_moments . . . 203
85.5 skewrnd . . . 203
85.6 skewrnd2 . . . 204

86 statistics **204**
86.1 range . . . 204
86.2 resample_with_replacement . . . 204

87 statistics/resampling-statistics/@Jackknife **204**
87.1 Jackknife . . . 204
87.2 estimated_STATIC . . . 205
87.3 matrix1_STATIC . . . 205
87.4 matrix2 . . . 205

88 statistics/resampling-statistics **205**
88.1 block_jackknife . . . 205
88.2 jackknife_moments . . . 205
88.3 moving_block_jackknife . . . 205
88.4 randblockserr . . . 206
88.5 resample . . . 206

89 statistics **206**
89.1 scale_quantile_sd . . . 206
89.2 sd_sample_quantiles . . . 206
89.3 skewpdf . . . 207
89.4 trimmed_mean . . . 207
89.5 ttest2_man . . . 207
89.6 ttest_man . . . 207
89.7 ttest_paired . . . 207

89.8 wgeomean . . . 207
89.9 wgeovar . . . 208
89.10 wharmean . . . 208
89.11 wharstd . . . 208
89.12 wharvar . . . 208

90 mathematics **208**
90.1 ternary_diagram . . . 208

91 test/master **208**
91.1 dat_test_lanczos_3d_k_20_n_40 . . . 208
91.2 poisson2d_blk . . . 208
91.3 qr_implicit_givens_2 . . . 208
91.4 spectral_derivative_2d . . . 209
91.5 test_2d_eigensolver_hydrogen . . . 209
91.6 test_2d_refine . . . 209
91.7 test_3d_eigensolver_hydrogen . . . 209
91.8 test_FEM . . . 209
91.9 test_Mesh_3d . . . 209
91.10 test_arnoldi . . . 209
91.11 test_arpackc . . . 209
91.12 test_assemble . . . 209
91.13 test_assembly_performance . . . 209
91.14 test_bc_one_sided . . . 210
91.15 test_compare_solvers . . . 210
91.16 test_complete . . . 210
91.17 test_convergence . . . 210
91.18 test_convergence_b . . . 210
91.19 test_df_2d . . . 210
91.20 test_eig_algs . . . 210
91.21 test_eig_inverse . . . 210
91.22 test_eigs_lanczos . . . 210
91.23 test_eigs_lanczos_1 . . . 210
91.24 test_eigs_lanczos_2 . . . 211
91.25 test_eigs_lanczos_performance . . . 211
91.26 test_fdm . . . 211
91.27 test_fdm_d_vargrid . . . 211
91.28 test_fdm_spectral . . . 211
91.29 test_fem . . . 211
91.30 test_fem_1d . . . 211
91.31 test_fem_1d_higher_order . . . 211
91.32 test_fem_2d_adaptive . . . 211
91.33 test_fem_2d_higher_order . . . 211
91.34 test_fem_3d_higher_order . . . 212

91.35 test_fem_3d_refine 212
91.36 test_fem_b 212
91.37 test_fem_derivative 212
91.38 test_fem_quadrature 212
91.39 test_final 212
91.40 test_fix_substitution 212
91.41 test_forward 212
91.42 test_get_sparse_arrays 212
91.43 test_harmonic_oscillator 212
91.44 test_high_order_fdm_periodic_bc 213
91.45 test_hydrogen_wf 213
91.46 test_ichol 213
91.47 test_interpolation 213
91.48 test_inverse_problem 213
91.49 test_it_vs_exact 213
91.50 test_jama 213
91.51 test_jd 213
91.52 test_jdqz 213
91.53 test_lanczos_2 213
91.54 test_lanczos_biorthogonal 214
91.55 test_laplacian 214
91.56 test_laplacian_non_uniform 214
91.57 test_laplacian_simple 214
91.58 test_mesh_2d_uniform 214
91.59 test_mesh_2d_uniform_2 214
91.60 test_mesh_circle 214
91.61 test_mesh_generation 214
91.62 test_mesh_interpolate 214
91.63 test_mg 214
91.64 test_minres_recycle 215
91.65 test_multigrid 215
91.66 test_nc 215
91.67 test_nonuniform_symmetric 215
91.68 test_pde 215
91.69 test_permutation 215
91.70 test_poison_fem 215
91.71 test_polar 215
91.72 test_potential 215
91.73 test_powers 215
91.74 test_precondition 216
91.75 test_project_rectangle 216
91.76 test_qr 216
91.77 test_quantum_well 216
91.78 test_radial_adaptive 216

91.79 test_radial_confinement 216
91.80 test_radial_fixes 216
91.81 test_refine_2d 216
91.82 test_refine_2d_b 216
91.83 test_refine_3d 216
91.84 test_refine_structural 217
91.85 test_regularisation 217
91.86 test_round_off 217
91.87 test_schrdinger_potentials 217
91.88 test_uniform_mesh 217
91.89 test_vargrid 217

92 test 217
92.1 test_gaussfit3 217
92.2 test_mtimes3x3 217

93 wavelet 218
93.1 contiuous_wavelet_transform 218
93.2 cwt_man 218
93.3 example_wavelets 218
93.4 phasewrap 218
93.5 test_cwt_man 218
93.6 test_phasewrap 218
93.7 test_wavelet 218
93.8 test_wavelet2 218
93.9 test_wavelet_analysis 219
93.10 test_wavelet_reconstruct 219
93.11 test_wtc 219
93.12 wavelet 219
93.13 wavelet_reconstruct 219
93.14 wavelet_transform 219

94 mathematics 219
94.1 wrapphase 219

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

re(c1 exp(io1x))*re(c2 exp(io2x)) =
      1/2*(    real(c1*c2*exp(i*(n1+n2)*o*x)) ...
            + 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 croots

```
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 exp10

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 leat 2*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 confidnce 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 2pi

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 2pi

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
    ti
```

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 continous fourier transform
this is a matrix with (2*n+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

powers of a^p = (ur + u1 sin(ot) + u2 sin(ot+dp))^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 = (ur + u1 sin(ot) + u2 sin(ot+dp))^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(jot) + b_j cos(jot))
  = Real(sum_{i=0}^inf c_i exp(1i*omega), c_i = a_i + b_i

NOT the alternative sum_{i=-inf}^inf \tilde c_i, tile c_j = 1/2 a_j
     + 1/2i b_j
```

11.28 fourier_predict

```
expand a continous fourier series at times t
```

11.29 fourier_quadratic_interaction_coefficients

11.30 fourier_range

```
approximate range of a continous Fourier series with 2 components
range(y) = max(y) - min(y)
```

11.31 fourier_regress

```
fit a continous 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 freqency 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 disctrete fourier transform

rule for peaks: there is no higher value left or right of the "peak
    "
              until the signal drops to p*y_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=^n a_i cos(j x) + b_i 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 geometry/@Geometry

12.1 Geometry

12.2 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
```

12.3 arclength_old

```
arc length of a two dimensional function
```

12.4 arclength_old2

```
arc length of a two dimensional function
```

12.5 base_point

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

12.6 base_point_limited

```
base point (Fusspunkt) of a point on a line
```

12.7 centroid

```
centroid pf a polygone
```

12.8 cosa_min_max

12.9 cross2

```
cross product in two dimensions
```

12.10 curvature

```
curvature of a function in two dimensions
```

12.11 ddot

```
sum of squares of cos of inner angles of triangle
```

12.12 distance

```
equclidan distance between two points
```

12.13 distance2

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

12.14 dot

```
dot product
```

12.15 edge_length

```
edge length
```

12.16 enclosed_angle

```
angle enclosed between two lines
```

12.17 enclosing_triangle

```
smallest enclosing equilateral triangle with bottom site paralle to
     X axis
```

12.18 hexagon

```
coordinates of a hexagon, scaled and rotated
```

12.19 inPolygon

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

12.20 inTetra

```
flag points contained in tetrahedron
```

12.21 inTetra2

```
flag points contained in tetrahedron
```

12.22 inTriangle

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

12.23 intersect

```
intersect between two lines
```

12.24 lineintersect

```
intersect of two lines
```

12.25 lineintersect1

```
intersect of two lines
```

12.26 minimum_distance_lines

```
minimum distance of two lines in three dimensions
```

12.27 mittenpunkt

```
mittenpunkt of a triangle
```

12.28 nagelpoint

```
nagelpoint of a triangle
```

12.29 onLine

12.30 orthocentre

```
orthocentre of triangle
```

12.31 plumb_line

12.32 poly_area

```
area of a polygon
function A = poly_area(x,y)
```

12.33 poly_edges

```
edges of a polygon
```

12.34 poly_set

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

12.35 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
```

12.36 polyxpoly

```
intersections of two polygons
```

12.37 project_to_curve

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

12.38 quad_isconvex

12.39 random_disk

```
draw random points on the unit disk
```

12.40 random_simplex

```
random point inside of a triangle
```

12.41 sphere_volume

```
volume of a sphere
```

12.42 tetra_volume

```
volume of a tetrahedron
```

12.43 tobarycentric

```
cartesian to barycentric coordinates
```

12.44 tobarycentric1

```
cartesian to barycentric coordinates
```

12.45 tobarycentric2

cartesian to barycentric coordinates

12.46 tobarycentric3

cartesian to barycentric coordinates

12.47 tri_angle

cos of angles of a triangle

12.48 tri_area

angle of a triangle

12.49 tri_centroid

centroid of a triangle

12.50 tri_distance_opposit_midpoint

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

12.51 tri_edge_length

edge length of a triangle

12.52 tri_edge_midpoint

mid point of a triangle

12.53 tri_excircle

```
excircle of a triangle
```

12.54 tri_height

```
height of a triangle
```

12.55 tri_incircle

```
incircle of a triangle
```

12.56 tri_isacute

```
flag acute triangles
```

12.57 tri_isobtuse

```
flag obntuse triangles
```

12.58 tri_semiperimeter

```
semiperimeter of a triangle
```

12.59 tri_side_length

```
edge lenght of triangle
```

13 geometry

13.1 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
```

13.2 bounding_box

```
bounding box of X
```

13.3 curvature_1d

```
curvature of a sampled parametric curve in two dimensions
```

13.4 cvt

```
centroidal voronoi tesselation
```

13.5 deg_to_frac

```
degree, minutes and seconds to fractions
```

13.6 ellipse

```
n-points on an ellipse
```

13.7 ellipseX

```
x-coordinates of y-coordinates of an ellipse
```

13.8 ellipseY

13.9 first_intersect

```
get first intersection between lines in A and B
```

13.10 golden_ratio

```
golden ratio
```

13.11 hypot3

```
hypothenuse in 3D
```

13.12 meanangle

```
weighted mean of angles
```

13.13 meanangle2

```
mean angle
```

13.14 meanangle3

```
mean angle
```

13.15 meanangle4

```
mean angle
```

13.16 medianangle

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

13.17 medianangle2

```
median angle

input
alpha : x*m, [rad] angle

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

13.18 pilim

```
limit to +- pi
```

13.19 streamline_radius_of_curvature

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

14 histogram/@Histogram

14.1 2x

14.2 Histogram

14.3 bimodes

14.4 cdf

14.5 cdfS

14.6 chi2test

14.7 cmoment

14.8 cmomentS

14.9 entropy

14.10 entropyS

14.11 iquantile

14.12 kstest

14.13 kurtosis

14.14 kurtosisS

14.15 mean

14.16 meanS

14.17 median

14.18 medianS

14.19 mode

14.20 modeS

14.21 moment

14.22 momentS

14.23 pdf

14.24 quantile

14.25 quantileS

14.26 setup

14.27 skewness

14.28 skewnessS

14.29 stairs

14.30 stairsS

14.31 std

14.32 stdS

14.33 var

14.34 varS

15 histogram

15.1 hist_man

15.2 histadapt

15.3 histconst

15.4 pdf_poly

15.5 plotcdf

15.6 test_histogram

16 linear-algebra

16.1 averaging_matrix_2

16.2 colnorm

```
norms of columns
```

16.3 condest_

```
estimation of the condition number
```

17 linear-algebra/coordinate-transformation

17.1 barycentric2cartesian

```
barycentric to cartesian coordinates
```

17.2 barycentric2cartesian3

```
convert barycentric to cartesian coordinates
```

17.3 cartesian2barycentric

```
cartesian to barycentric coordinates
```

17.4 cartesian_to_unit_triangle_basis

```
transform coodinates into unit triangle
```

17.5 ellipsoid2geoid

17.6 example_approximate_utm_conversion

17.7 latlon2utm

```
transform latitude and longitude to WGS84 UTM
```

17.8 latlon2utm_simple

17.9 lowrance_mercator_to_wgs84

```
convert lowrance coordinates to wgs84

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

17.10 nmea2utm

```
convert nmea messages to utm coordinates
```

17.11 sn2xy

```
convert sn to xy coordinates
```

17.12 unit_triangle_to_cartesian

```
transform coordinates in unit triangle to cartesian coordinates
```

17.13 utm2latlon

```
convert wgs84 utm to latitute and longitude
```

17.14 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
```

17.15 xy2sn

```
convert cartesian to streamwise coordiantes
```

17.16 xy2sn_java

```
use java port for speed up
```

17.17 xy2sn_old

```
transform points from cartesian into streamwise coordinates

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

18 linear-algebra

18.1 det2x2

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

18.2 det3x3

```
determinant of stacked 3x3 matrices
```

18.3 det4x4

```
determinant of stacked 4x4 matrices
```

18.4 diag2x2

```
diagonal of stacked 2x2 matrices
```

18.5 eig2x2

```
eigenvalues of stacked 2x2 matrices
```

19 linear-algebra/eigenvalue

19.1 eig_bisection

19.2 eig_inverse

19.3 eig_inverse_iteration

19.4 eig_power_iteration

20 linear-algebra/eigenvalue/jacobi-davidson

20.1 afun_jdm

20.2 davidson

20.3 jacobi_davidson

20.4 jacobi_davidson_qr

20.5 jacobi_davidson_qz

20.6 jacobi_davidson_simple

20.7 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=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,rnrm/snrm]];
% 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'
```

20.8 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=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=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,rnrm/snrm]];
% 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'
```

20.9 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=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
    ==================================
% accepted separation between eigenvalues:
% no preconditioning
% solve left preconditioned system
% compute vectors and matrices for skew projection
% precondion and project r
% solve preconditioned system
% no preconditioning
% solve two-sided expl. precond. system
% compute vectors and matrices for skew projection
% precondion 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
```

20.10 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 precond=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,rnrm/snrm]];
% 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,rnrm/snrm]];
% 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
    ===========================================
%======================================================================

%======================================================================

%======================================================================

%======================================================================
```

20.11 mfunc_jdm

20.12 mgs

20.13 minres_

20.14 mv_jacobi_davidson

21 linear-algebra

21.1 first

21.2 gershgorin_circle

```
range of eigenvalues determined by the gershgorin circle theorem
```

21.3 haussdorff

```
haussdorf dimension
box counting: count cectangles 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
```

21.4 ieig2x2

```
reconstruct matrix from eigenvalue decomposition
```

21.5 inv2x2

```
2x2 inverse of stacked matrices
```

21.6 inv3x3

21.7 inv4x4

```
inverse of stacked 4x4 matrices
```

22 linear-algebra/lanczos

22.1 arnoldi

22.2 arnoldi_new

22.3 eigs_lanczos_man

22.4 lanczos

22.5 lanczos_

22.6 lanczos_biorthogonal

22.7 lanczos_biorthogonal_improved

22.8 lanczos_ghep

22.9 mv_lanczos

22.10 reorthogonalise

22.11 test_lanczos

23 linear-algebra/linear-systems

23.1 gmres_man

```
break on convergence
```

23.2 minres_recycle

24 linear-algebra

24.1 lpmean

```
mean of pth-power of a
```

24.2 lpnorm

```
norm of lth-power of a
```

24.3 matvec3

```
matrix-vector product of stacked matrices and vectors
```

24.4 max2d

```
maximum value and i-j index for matrix
```

24.5 mpoweri

```
approximation of A^p, where p is not integer by quadtratic
    interpolation
```

24.6 mtimes2x2

24.7 mtimes3x3

```
product of stacked 3x3 matrices
```

24.8 nannorm

```
norm of a vector, skips nan-values
```

24.9 nanshift

```
shift vector, but set out of range values to NaN
```

24.10 nl

```
number rows (lines) of a matrix

analogue to unix nl command
```

24.11 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)
```

24.12 normalize1

```
normalize columns in x to [-1,1]
```

24.13 normrows

24.14 orth2

```
make matrix A orhogonal to B
```

24.15 orth_man

```
orthogonalize the columns of A
```

24.16 orthogonalise

```
make x orthogonal to Y
```

24.17 paddext

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

24.18 paddval1

```
padd values at end of x
```

24.19 paddval2

```
padd values to x
```

25 linear-algebra/polynomial

25.1 chebychev

```
chebycheff polynomials
```

25.2 piecewise_polynomial

```
evaluate piecewise polynomial
```

25.3 roots1

```
roots of linear functions
```

25.4 roots2

```
roots of quadratic function
c1 x^2 + c2 x + c3 = 0
```

25.5 roots2poly

25.6 roots3

25.7 roots4

25.8 test_roots4

25.9 vanderi_1d

```
vandermonde matrix of an integral
```

26 linear-algebra

26.1 randrot

```
random rotation matrix
```

26.2 right

```
get right column by shifting columns to left
extrapolate rightmost column
```

26.3 rot2

```
rotation matrix from angle
```

26.4 rot2dir

```
rotation matrix from direction vector
```

26.5 rot3

26.6 rotR

26.7 rownorm

26.8 simmilarity_matrix

26.9 spnorm

```
frobenius norm
```

26.10 spzeros

```
allocate a sparze matrix of zeros
```

26.11 test_roots3

26.12 transform_minmax

26.13 transpose3

```
transpose stacked 3x3 matrices
```

26.14 transposeall

27 logic

```
bitwise operations on integers
```

27.1 bitor_man

```
 bitwise OR of the numbers of the columns of A

 input:
       A (positive integer)
```

28 master/plot

28.1 attach_boundary_value

28.2 cartesian_polar

28.3 img_vargrid

28.4 plot_basis_functions

28.5 plot_convergence

28.6 plot_dof

28.7 plot_eigenbar

28.8 plot_error_estimation

28.9 plot_error_estimation_2

28.10 plot_error_fem

28.11 plot_fdm_kernel

28.12 plot_fdm_vs_fem

28.13 plot_fem_accuracy

28.14 plot_function_and_grid

28.15 plot_hat

28.16 plot_hydrogen_wf

28.17 plot_mesh

28.18 plot_mesh_2

28.19 plot_refine

28.20 plot_refine_3d

28.21 plot_runtime

28.22 plot_spectrum

28.23 plot_wavefunction

29 master/ported

29.1 assemble_2d_dphi_dphi

29.2 assemble_2d_phi_phi

29.3 assemble_3d_dphi_dphi

29.4 assemble_3d_phi_phi

29.5 dV_2d_

29.6 derivative_2d

29.7 derivative_3d

29.8 element_neighbour_2d

29.9 prefetch_2d_

29.10 promote_2d_3_10

29.11 promote_2d_3_15

29.12 promote_2d_3_21

29.13 promote_2d_3_6

29.14 promote_3d_4_10

29.15 promote_3d_4_20

29.16 promote_3d_4_35

29.17 vander_2d

29.18 vander_3d

30 master/sandbox

30.1 adapt

30.2 assoc_laguerre

30.3 assoc_legendre

30.4 c23

31 master/sandbox/cg

31.1 cg

31.2 cg_coef_to_poly

31.3 errmat

31.4 lanczos

31.5 laplacian_2d

31.6 test_cg_eigs

31.7 test_lanczos

32 master/sandbox

32.1 condition_number_higher_order

32.2 confinement_dat

32.3 convergence_2d_3d

32.4 convergence_matrix_powers

32.5 cut_out

32.6 derivative_2d

32.7 derivative_3d

32.8 dummy

32.9 eig_error

32.10 eigs_fix

32.11 energy_level

32.12 equalise

32.13 example_int64

```
Basic operations
```

```
Matrix multiplication
Timing
```

33 master/sandbox/fem-matlab

33.1 boundary_circle

33.2 boundary_rectangle

33.3 geometry_circle_with_hole

33.4 geometry_rectangle

34 master/sandbox

34.1 fem_2d_estimate_error

34.2 fem_assemble_scratch

34.3 fem_s

34.4 fourier_h

34.5 grad_2d

34.6 grad_3d

34.7 gradient

34.8 harmonic_oscillator

34.9 hydrogen_2d_analytic

34.10 hydrogen_boxed

34.11 hydrogen_boxed_old

34.12 hydrogen_wave

```
% Hydrogen atom
```

34.13 hydrogen_wf

34.14 ichol_man

34.15 known_eigenvalue

34.16 kron_man

34.17 laguerre

34.18 laplacian_arbitrary_order_old

34.19 laplacian_convergence

34.20 laplacian_cut_out

34.21 laplacian_cylindrical

34.22 laplacian_non_uniform_old

34.23 laplacian_polar

34.24 laplacian_simple

34.25 lderivative_3d

34.26 list_dat

34.27 matlab-horner

34.28 mesh_to_grid_2d_3

34.29 mg_mat

34.30 mv

34.31 orth2

34.32 partial_derivative_2d

34.33 partition_function

34.34 partition_function_old

34.35 poisson

34.36 poisson_fem

34.37 potential

34.38 powerc

34.39 quick_newihbour

34.40 radial

34.41 radial_convergence

34.42 radial_wafefunction

34.43 refine_2d

34.44 refine_3d

34.45 relerr

34.46 restore_cw

34.47 runtime_bm

34.48 rydberg

34.49 s_old

34.50 snorm

34.51 spherical_harmonic

34.52 split_eig

34.53 sum1

34.54 sum3

35 master/sandbox/summation

35.1 acc

35.2 add

35.3 ape

35.4 mmul_accurately

35.5 sum_kahan

35.6 sum_pairwise

35.7 test_sum

36 master/sandbox

36.1 test_convergence_ill_conditioned

36.2 test_fem_1d

36.3 test_fem_2d

36.4 test_fem_3d

36.5 test_increase

36.6 test_lanczos_shift

36.7 test_ldl

36.8 test_power

36.9 trefethen_p8_fdm

36.10 wavefunc

36.11 xgrid

37 number-theory

37.1 ceiln

```
floor to leading n-digits
```

37.2 digitsb

```
number of digits with respect to specified base
```

37.3 floorn

```
floor to n-digits
```

37.4 iseven

```
true for even numbers in X
```

37.5 multichoosek

```
all combinations of lenght 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
```

37.6 nchoosek_man

```
vecotrised binomial coefficient
b = N!/K!(N-K)!
```

37.7 pythagorean_triple

```
pythagorean triple
```

37.8 roundn

```
round to n digits
```

38 numerical-methods/differentiation

38.1 derivative1

```
first derivative on variable mesh
second order accurate
```

38.2 derivative2

```
second derivative on a variable mesh
```

39 numerical-methods/finite-difference

39.1 cdiff

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

39.2 cdiffb

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

39.3 cmean

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

39.4 derivative_matrix_1_1d

```
finite difference matrix of first derivative in one dimensions
```

39.5 derivative_matrix_2_1d

```
finite derivative matrix of second derivative in one dimension
```

39.6 derivative_matrix_2d

```
finite difference derivative matrix in two dimensions
```

39.7 derivative_matrix_curvilinear

```
derivative matrix on a curvilinear grid
```

39.8 derivative_matrix_curvilinear_2

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

39.9 difference_kernel

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

39.10 distmat

```
distance matrix for a 2 dimensional rectangular matrix
```

39.11 gradpde2d

```
objective function gradiend 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
```

39.12 laplacian

39.13 laplacian_fdm

```
finite difference matrix of the laplacian
BC
```

39.14 left

```
left element of vector, leftmost column is extrapolated
```

39.15 lrmean

```
mean of the left and right element
```

40 numerical-methods/finite-difference/master

40.1 fdm_adaptive_grid

40.2 fdm_adaptive_refinement_old

40.3 fdm_assemble_d1_2d

40.4 fdm_assemble_d2_2d

40.5 fdm_confinement

40.6 fdm_d_vargrid

40.7 fdm_h_unstructured

40.8 fdm_hydrogen_vargrid

40.9 fdm_mark_unstructured_2d

40.10 fdm_plot

40.11 fdm_plot_series

40.12 fdm_refine_2d

40.13 fdm_refine_3d

40.14 fdm_refine_unstructured_2d

40.15 fdm_schroedinger_2d

40.16 fdm_schroedinger_3d

40.17 relocate

41 numerical-methods/finite-difference

41.1 mid

```
mid point between neighbouring vector elements
```

41.2 pwmid

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

41.3 ratio

```
ratio of two subsequent values
```

41.4 steplength

```
step length of a vector if it were equispaced
```

41.5 swapoddeven

```
swap odd and even elements in a vector
```

41.6 test_derivative_matrix_2d

41.7 test_derivative_matrix_curvilinear

41.8 test_difference_kernel

42 numerical-methods/finite-element

42.1 Mesh_2d_java

42.2 Tree_2d_java

42.3 assemble_1d_dphi_dphi

42.4 assemble_1d_phi_phi

42.5 assemble_2d_dphi_dphi_java

42.6 assemble_2d_phi_phi_java

42.7 assemble_3d_dphi_dphi_java

42.8 assemble_3d_phi_phi_java

42.9 boundary_1d

42.10 boundary_2d

42.11 boundary_3d

42.12 check_area_2d

42.13 circmesh

42.14 cropradius

42.15 display_2d

42.16 display_3d

42.17 distort

42.18 err_2d

42.19 estimate_err_2d_3

42.20 example_1d

42.21 example_2d

42.22 explode

42.23 fem_2d

42.24 fem_2d_heuristic_mesh

42.25 fem_get_2d_radial

42.26 fem_interpolation

42.27 fem_plot_1d

42.28 fem_plot_1d_series

42.29 fem_plot_2d

42.30 fem_plot_2d_series

42.31 fem_plot_3d

42.32 fem_plot_3d_series

42.33 fem_plot_confine_series

42.34 fem_radial

```
adaptive grid
constant grid
```

42.35 flip_2d

42.36 get_mesh_arrays

42.37 hashkey

43 numerical-methods/finite-element/int

43.1 int_1d_gauss

43.2 int_1d_gauss_1

43.3 int_1d_gauss_2

43.4 int_1d_gauss_3

43.5 int_1d_gauss_4

43.6 int_1d_gauss_5

43.7 int_1d_gauss_6

43.8 int_1d_gauss_lobatto

43.9 int_1d_nc_2

43.10 int_1d_nc_3

43.11 int_1d_nc_4

43.12 int_1d_nc_5

43.13 int_1d_nc_6

43.14 int_1d_nc_7

43.15 int_1d_nc_7_hardy

43.16 int_2d_gauss_1

43.17 int_2d_gauss_12

43.18 int_2d_gauss_13

43.19 int_2d_gauss_16

43.20 int_2d_gauss_25

43.21 int_2d_gauss_3

43.22 int_2d_gauss_33

43.23 int_2d_gauss_6

43.24 int_2d_gauss_7

43.25 int_2d_gauss_9

43.26 int_2d_nc_10

43.27 int_2d_nc_15

43.28 int_2d_nc_21

43.29 int_2d_nc_3

43.30 int_2d_nc_6

43.31 int_3d_gauss_1

43.32 int_3d_gauss_11

43.33 int_3d_gauss_14

43.34 int_3d_gauss_15

43.35 int_3d_gauss_24

43.36 int_3d_gauss_4

43.37 int_3d_gauss_45

43.38 int_3d_gauss_5

43.39 int_3d_nc_11

43.40 int_3d_nc_4

43.41 int_3d_nc_6

43.42 int_3d_nc_8

44 numerical-methods/finite-element

44.1 interpolation_matrix

44.2 mark

44.3 mark_1d

44.4 mesh_1d_uniform

44.5 mesh_3d_uniform

44.6 mesh_interpolate

44.7 neighbour_1d

44.8 old

44.9 pdeeig_1d

44.10 pdeeig_2d

44.11 pdeeig_3d

44.12 polynomial_derivative_1d

44.13 potential_const

44.14 potential_coulomb

44.15 potential_harmonic_oscillator

44.16 project_circle

44.17 project_rectangle

44.18 promote_1d_2_3

44.19 promote_1d_2_4

44.20 promote_1d_2_5

44.21 promote_1d_2_6

44.22 quadrilaterate

44.23 recalculate_regularity_2d

44.24 refine_1d

44.25 refine_2d_21

44.26 refine_2d_structural

44.27 regularity_1d

44.28 regularity_2d

44.29 regularity_3d

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

44.30 relocate_2d

44.31 test_circmesh

44.32 test_hermite

44.33 tri_assign_points

44.34 triangulation_uniform

44.35 vander_1d

```
 van der Monde matrix
```

44.36 vanderd_1d

44.37 vanderi_1d

45 numerical-methods/finite-volume/@Advection

45.1 Advection

```
FVM treatment of the Advection equation
```

45.2 dot_advection

```
advection equation
```

46 numerical-methods/finite-volume/@Burgers

46.1 burgers_split

```
viscous Burgers' equation,
mixed analytic and numerical derivative in frequency space
by splitting sheme
u_t = -(0.5*u^2)_x + c*u_xx
```

46.2 dot_burgers_fdm

```
viscous burgers' equation
u_t = -d/dx (1/2*u^2) + c d^2/dx^2 u_xx
```

46.3 dot_burgers_fft

```
viscous Burgers' equation in frequency space
u_t + (0.5*u^2)_x = c*u_xx
```

47 numerical-methods/finite-volume/@Finite_Volume

47.1 Finite_Volume

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

47.2 apply_bc

```
apply boundary conditions
```

47.3 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);
```

47.4 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
```

47.5 step_unsplit

```
step in time, without splitting the inhomogeneous term
```

48 numerical-methods/finite-volume/@Flux_Limiter

48.1 Flux_Limiter

```
class of flux limiters
```

48.2 beam_warming

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

48.3 fromm

```
fromme limiter
low res
```

48.4 lax_wendroff

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

48.5 minmod

```
min-mod schock limiter
```

48.6 monotized_central

```
monotonized central flux limiter
```

48.7 muscl

```
muscl flux limiter
```

48.8 superbee

```
superbee limiter
```

48.9 upwind

```
godunov scheme
godunov, first order accurate
```

48.10 vanLeer

```
van Leer limiter
```

49 numerical-methods/finite-volume/@KDV

49.1 dot_kdv_fdm

```
korteweg de vries equation
u_t + (0.5*u^2)_x = c*u_xxx
```

49.2 dot_kdv_fft

```
korteweg de vries equation
compute derivatives in frequency space
u_t + (0.5*u^2)_x = c*u_xxx
```

49.3 kdv_split

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

50 numerical-methods/finite-volume/@Reconstruct_Average_E

50.1 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;
```

50.2 advect_highres

```
single time step for the reconstruct evolve algorithm
```

50.3 advect_lowress

```
single time step
low resolution
```

51 numerical-methods/finite-volume

51.1 Godunov

```
Godunov, upwind method for systems of pdes
```

51.2 Lax_Friedrich

```
Lax-Friedrich-Method
for hyperbolic conservation laws
err = O(dt) + O(dx)
|a dt/dx| < 1
```

51.3 Measure

51.4 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
```

51.5 fv_swe

```
wrapper for solving SWE
```

51.6 staggered_euler

```
forward euler method with staggered grid
```

51.7 staggered_grid

```
staggered grid approximation to the SWE
```

52 numerical-methods

52.1 grid2quad

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

53 numerical-methods/integration

53.1 cumintL

```
cumulative integral from left to right
```

53.2 cumintR

```
cumulative integral from right to left
```

53.3 int_trapezoidal

```
integrate y along x with the trapezoidal rule
```

54 numerical-methods/interpolation/@Kriging

54.1 Kriging

```
class for Kriging interpolation
```

54.2 estimate_semivariance

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

54.3 interpolate_

```
interpolate with Krieging 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  : targe point coordinates
Vt  : value at target points
E2t : squared interpolation error at target points
```

55 numerical-methods/interpolation/@RegularizedInterpolator1

55.1 RegularizedInterpolator1

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

55.2 init

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

56 numerical-methods/interpolation/@RegularizedInterpolator

56.1 RegularizedInterpolator2

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

56.2 init

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

57 numerical-methods/interpolation/@RegularizedInterpolator

57.1 RegularizedInterpolator3

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

57.2 init

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

58 numerical-methods/interpolation

58.1 IDW

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

58.2 IPoly

```
polynomial interpolation class
```

58.3 IRBM

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

58.4 ISparse

```
sparse interpolation class
```

58.5 Inn

```
nearest neighbour interpolation
```

58.6 Interpolator

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

58.7 fixnan

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

58.8 idw1

```
spatial average ny inverse distance weighting
```

58.9 idw2

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

58.10 inner2outer

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

58.11 inner2outer2

```
interpolate from element (segment) centres to edge points
```

58.12 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
```

58.13 interp1_man

```
interpolate
```

58.14 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
```

58.15 interp1_slope

```
quadratic interpolation returning value and derivative(s)
```

58.16 interp1_smooth

58.17 interp1_unique

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

58.18 interp2_man

```
nearest neighbour interpolation in two dimensions
```

58.19 interp_angle

```
interpolate an angle
```

58.20 interp_fourier

```
interpolation by the fourier method
```

58.21 interp_fourier_batch

```
batch interpolation by the fourier interpolation
```

58.22 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_);
```

58.23 interp_sn2

```
interpolation in streamwise coordinates
```

58.24 interp_sn3

58.25 interp_sn_

58.26 limit_by_distance_1d

```
smooth subsequent values along a curve such that
      v(x0+dx) < v(x0) + (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)
```

58.27 resample1

```
interpolation along a parametric curve with variable step width
```

58.28 resample_d_min

```
resample a function
```

58.29 resample_vector

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

58.30 test_interp1_limited

59 numerical-methods

59.1 inverse_complex

60 numerical-methods/ode

60.1 bvp2_check_arguments

60.2 bvp2c

```
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])
```

60.3 bvp2c2

```
solve second order boundary value problem via roots of the
    characteristic
polynomial

input:

x   : [nx1] discretized domain
      n : number of vertices
     nxc = n-1 : number of segments
```

```
bc  : struct : boundary condition
      bc.p(1)*y(0) + bc.pd(2)*y'(0) = bc.val(1)
      bc.p(2)*y(L) + bc.pd(2)*y'(L) = bc.val(2)

output:

A   : [2*nxc x 2*ns] disrcretisation matrix
rhs : [2*nxc x 1] right hand size

y = A^-1 rhs
```

60.4 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])
```

60.5 bvp2wavetrain

```
solve second order boundary value problem by repeated integration
```

60.6 bvp2wavetwopass

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

60.7 ivp_euler_forward

```
solve intial value problem by the euler forward method
```

60.8 ivprk2

```
solve initial value problem by the two step runge kutta method
```

60.9 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]
```

60.10 ode2characteristic

```
second order odes
transmittded and reflected wave
```

60.11 step_trapezoidal

```
single trapezoidal step
```

60.12 test_bvp2

61 numerical-methods/optimisation

61.1 armijo_stopping_criterion

```
armijo stopping criterion for optimizations
```

61.2 astar

```
astar path finding alforithm
```

61.3 binsearch

```
binary search on a line
```

61.4 bisection

```
bisection
```

61.5 box1

```
test objective function for optimisation routines
```

61.6 box2

61.7 cauchy

61.8 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
```

61.9 directional_derivative

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

61.10 dud

```
optimization by the dud algorithm
```

61.11 extreme3

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

61.12 extreme_quadratic

61.13 ftest

61.14 fzero_bisect

61.15 fzero_newton

61.16 grad

```
numerical gradient
```

61.17 hessian

```
numerical hessian
```

61.18 hessian_from_gradient

```
numerical hessian from gradient
```

61.19 hessian_projected

```
numerical hessian projected to one dimenstion
```

61.20 line_search

```
bisection routine
```

61.21 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
```

61.22 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
```

61.23 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
```

61.24 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
```

61.25 line_search_quadratic2

```
quadratic line search
```

61.26 line_search_wolfe

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

61.27 ls_bgfs

```
least squares by the bgfs method
```

61.28 ls_broyden

```
least squares by the broyden method
for rectangular / non symmetric systems
      Numerical Optimization nocedal
      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
```

61.29 ls_generalized_secant

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

61.30 nlcg

```
non-linear conjugate gradient
input:
x   : nx1 start vectort
```

```
opt : struct options
fdx : gradient constraint
```

61.31 nlls

```
non-linear least squares
```

61.32 picard

```
picard iteration
```

61.33 poly_extrema

```
extrema of a polynomial
```

61.34 quadratic_function

```
evaluate quadratic function in higher dimensions
```

61.35 quadratic_programming

```
optimize by quadratic programming
```

61.36 quadratic_step

```
single step of the quadratic programming
```

61.37 rosenbrock

```
rosenbrock test function
```

61.38 sqrt_heron

Heron's method for the square root

61.39 test_directional_derivative

61.40 test_dud

61.41 test_fzero_newton

61.42 test_line_search_quadratic2

61.43 test_ls_generalized_secant

61.44 test_nlcg_6_order

61.45 test_nlls

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

62 numerical-methods/pde

62.1 laplacian2d_fundamental_solution

63 numerical-methods/piecewise-polynomials

63.1 Hermite1

```
hermite polynomial interpolation in 1d
```

63.2 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)
```

63.3 hp2_predict

```
prediction with pw hermite polynomial
c are values at support points
```

63.4 hp_predict

```
predict with piecewise hermite polynomial
```

63.5 hp_regress

```
fit piecewise hermite polynomial
coefficients are values and derivatives
```

63.6 lp_count

```
lagrangian basis for interpolation
count number of valid samples
```

63.7 lp_predict

```
lagrangian basis piecwie interpolation, predicor
```

63.8 lp_regress

63.9 lp_regress_

64 regression/@PolyOLS

64.1 PolyOLS

```
class for polynomial least squares
```

64.2 coefftest

64.3 detrend

```
detrending by polynomial regression
```

64.4 fit

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

64.5 fit_

```
fit a polynomial function
```

64.6 predict

```
predict polynomial function values
```

64.7 predict_

64.8 slope

```
slope by linear regression
```

65 regression/@PowerLS

65.1 PowerLS

```
class for power law regression
```

65.2 fit

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

65.3 predict

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

65.4 predict_

66 regression/@Theil

66.1 Theil

```
Kendal-Theil-Sen robust regression
```

66.2 detrend

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

66.3 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
```

66.4 predict

```
predict values and confidence intervals with the Theil-Sen method
```

66.5 slope

```
fit the slope with the Theil-Sen method
```

67 regression

```
linear and non-linear regression
```

67.1 Theil_Multivariate

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

67.2 areg

```
regression using the pth-fraction of samples with smallest residual
```

67.3 ginireg

```
gini regression
```

67.4 hesssimplereg

```
hessian, gradient and objective function value of the simple
    regression
rhs = p(1) + p(2) x + eps
```

67.5 l1lin

```
solve ||Ax - b||_L1 by means of linear programming
```

67.6 lsq_sparam

```
parameter covariance of the least squares regression

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

67.7 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
```

67.8 regression_method_of_moments

```
fit linear function ||a b x = y||_L2 by the method of moments
y+eps = alpha + beta*x
```

67.9 robustlinreg

```
fit a linear function by splitting the x-values at their median
      (med(y_left) - med(y_right))/(med(x_left)-med(x_right)
this approach performs poorly compared to the theil-senn operator
```

67.10 theil2

```
Theil senn-estimator for two dimensions (glm)
```

67.11 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
```

67.12 total_least_squares

```
total least squares
```

67.13 weighted_median_regression

```
weighted median regression
c.f. Scholz, 1978
```

68 set-theory

68.1 issubset

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

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

69 signal-processing

69.1 acf_effective_sample_size

```
effective sample size from acf
```

69.2 acf_genton

```
autocorrelation function
```

69.3 acfar1

```
Autocorrelation function of the finite AR1 process

a_k = 1/(n-k)sum x_ix_i+1 + (xi + xi+k)mu + mu^2
    = r^k + 1/n sum_ij + 1/n
       pause
```

69.4 acfar1_2

```
autocorrelation of the ar1 process
```

69.5 acfar2

```
impulse response of the ar2 process
```

69.6 acfar2_2

```
autocorrelation of the ar2 process
X_i + a1 X_i-1 + a2 X_i-2 = 0
```

69.7 ar1_cutoff_frequency

69.8 ar1_effective_sample_size

```
effective sample size correction for autocorrelated series
```

69.9 ar1_mse_mu_single_sample

```
standard error of a single sample of an ar1 correlated process
```

69.10 ar1_mse_pop

```
variance of the population mean of a single realisation around zero

E[(mu_N-0)^2] = E[mu_N^]
```

69.11 ar1_mse_range

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

69.12 ar1_spectrum

```
spectrum of the ar1 process
```

69.13 ar1_to_tikhonov

```
convert ar1 correlation to tikhonovs lambda
```

69.14 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
```

69.15 ar1_var_factor_

```
variance of an autocorrelated finite process
```

69.16 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|
```

69.17 ar1delay

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

69.18 ar1delay_old

```
autocorrelation of the residual
```

69.19 ar2conv

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

69.20 ar2dof

```
effective samples size for the ar2 process
```

69.21 ar2param

```
ar2 parameter estimation from first two terms of acf

acf = [1 a1 a2 ...]
```

69.22 asymwin

```
creates asymmetrical filter windows
filter will always have negative weights
```

69.23 autocorr_fft

```
autocorrelation function
```

69.24 bandpass

```
bandpass filter
```

69.25 bandpass2

```
bandpass filter
```

69.26 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
```

69.27 bartlett_spectrogram

```
bartlet spectrogramm
TODO sliding window
```

69.28 bin1d

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

69.29 bin2d

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

69.30 binormrnd

```
generate two correlated normally distributed vectors
```

69.31 conv1_man

```
convolutions with padding
```

69.32 conv2_man

```
convolution in 2d
```

69.33 conv2z

69.34 conv30

```
convolve with rectangular window of lenght n
circular boundaries
```

69.35 conv_

```
convolution of a with b
```

69.36 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 lenght of f odd at first
```

69.37 convz

69.38 cosexpdelay

69.39 csmooth

```
smooth recursively with [1,2,1]/4 kernel
```

69.40 daniell_window

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

69.41 danielle_window

```
danielle fourier window
```

69.42 db2neper

```
convert decibel to neper
```

69.43 db2power

```
power ratio from db
```

69.44 derive_danielle_weight

69.45 derive_limit_0_acfar

69.46 detect_peak

```
detect peaks in a vector
requires function value to fall to p*max before new value is
    allowed
```

69.47 digital_low_pass_filter

```
design coefficients of a low pass filter with specified cut of
    frequency
and sampling period
alalogue low pass with pole at s=-omega_c=1/tau=1/RC
Ha = tau/(tau + s) = 1/(1 + omega_c*s)
```

69.48 doublesum_ij

```
double sum of r^i
```

69.49 effective_sample_size_to_ar1

```
convert effective sample size to ar1 correlation
```

69.50 filt_hodges_lehman

69.51 filter1

```
filter along one dimension
```

69.52 filter2

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

69.53 filter_

```
invalidate values that exceed n-times the robust standard deviation
```

69.54 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 poisition in the colum (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 alising (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
```

69.55 filterp

69.56 filterp1

```
fir filter with some fancy extras
```

69.57 filterstd

69.58 firls_man

```
design finite impulse response filter by the least squares method
```

69.59 flattopwin

```
the flat top window
```

69.60 frequency_response_boxcar

```
frquency response of a boxcar filter
```

69.61 freqz_boxcar

```
frequncy response of a boxcar filter
```

69.62 gaussfilt1

```
filter data series with a gaussian window
```

69.63 hanchangewin

```
hanning window for change point detection
```

69.64 hanchangewin2

```
nanning window for chage point detection
```

69.65 hanwin

```
hanning filter window
```

69.66 hanwin_

```
hanning filter window
```

69.67 highpass

```
high pass filter
```

69.68 kaiserwin

```
kaiser filter window
```

69.69 kalman

```
Kalman filter
```

69.70 lanczoswin

```
Lanczos window
```

69.71 last

```
lake tail, but for matrices
```

69.72 lowpass

```
low pass filter
```

69.73 lowpass2

```
design low pass filter with cutoff-frequency f1
```

69.74 lowpass_iir

```
iir-low pass
```

69.75 lowpass_iir_symmetric

```
two-sided iir low pass filter (for symmetry)
```

69.76 lowpassfilter2

```
low-pass filter of data
```

69.77 maxfilt1

69.78 meanfilt1

```
moving average filter with special treatment of the boundaries
```

69.79 medfilt1_man

```
moving median filter, supports columnwise operation
```

69.80 medfilt1_man2

```
moving median filter with special treatment of boundaries
```

69.81 medfilt1_padded

```
median filter with padding
```

69.82 medfilt1_reduced

```
median filter with padding
```

69.83 mid_term_single_sample

```
variance of single sample, mid term
```

69.84 minfilt1

69.85 mu2ar1

```
error variance of the mean of the finite length ar1 process
```

```
(mu)^2 = (sum epsi)^2 = sum_i sum_j eps_i eps_j = sum_ii(rho,n)/n^2
this has the limit s^2 for rho->1
```

69.86 mysmooth

69.87 nanautocorr

```
autocorrelation with nan-values
```

69.88 nanmedfilt1

```
medfilt1, skipping nans
```

69.89 neper2db

```
convert neper to db
```

69.90 peaks_man

```
peaks of a periodogram
```

69.91 polyfilt1

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

69.92 qmedfilt1

```
medfilt1, after fitting a quadratic polynomial
```

69.93 randar1

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

69.94 randar1_dual

```
draw random variables of two corrlated ar1 processes
```

69.95 randar2

```
generate ar2 process
```

69.96 randarp

```
randomly generate the instance of an ar-p process
```

69.97 range_window

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

69.98 rectwin

```
rectangular window
```

69.99 recursive_sum

69.100 select_range

69.101 smooth1d_parametric

```
smooth position of p0=x0,y0 between p1=x1,y1 and p2=x2,y2,
so that distance to p1 and p2 becomes equal
and the chord length remains the same
```

69.102 smooth2

```
smooth vectos of X
```

69.103 smooth_man

69.104 smooth_parametric

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

69.105 smooth_parametric2

```
parametrically smooth the curve
```

69.106 smooth_with_splines

69.107 smoothfft

```
filter with fast fourier transform
```

69.108 spectrogram

```
spectrogram
```

69.109 std_window

```
moving block standard deviation
```

69.110 sum_i_lag

```
sum of ar1 matrix with lag
sum_i=1^n rho^|i-k|
```

69.111 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
```

69.112 sum_ii_

69.113 sum_ij

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

69.114 sum_ij_

69.115 sum_ij_partial_

69.116 sum_multivar

```
sum of matrix entries of bivariate ar1 process
```

69.117 test_acfar1

69.118 test_acfar1_2

69.119 test_acfar1_3

69.120 test_acfar1_4

69.121 test_acfar2

69.122 test_ar1_var_factor

69.123 test_ar1_var_factor_2

69.124 test_ar1_var_mu_single_sample

69.125 test_ar1_var_pop

69.126 test_ar1_var_pop_1

69.127 test_ar1delay

69.128 test_bivariate_covariance_term

69.129 test_convexity

69.130 test_lanczoswin

69.131 test_madcorr

69.132 test_randar1

69.133 test_randar1_multivariate

69.134 test_randar2

69.135 test_sum_ij

69.136 test_sum_multivar

69.137 test_trifilt1

69.138 test_wautocorr

69.139 test_wavelet_transform

69.140 test_wordfilt

69.141 test_xar1_mid_term

69.142 tikhonov_to_ar1

```
convert coefficient of the tikhonov regularization to correlatioon
    of the ar1 process
```

69.143 trapwin

```
trapezoidal filter window
```

69.144 trifilt1

```
filter with triangular window
```

69.145 triwin

```
triangular filter window
```

69.146 triwin2

```
triangular filter window
```

69.147 varar1

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

69.148 welch_spectrogram

```
welch spectrogram
```

69.149 wfilt

```
filter with window
```

69.150 winbandpass

```
filter with bandpass
```

69.151 window_make_odd

69.152 winfilt0

```
filter with window
```

69.153 winlength

```
window length for desired cutoff frequency
power at fc 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
```

69.154 wmeanfilt

```
mean filter with window
```

69.155 wmedfilt

```
median filter with window
```

69.156 wordfilt

```
weighted order filter
```

69.157 wordfilt_edgeworth

```
weighed order filter
```

69.158 xar1

69.159 xcorr_man

```
cross correlation of two sampled ar1 processes
```

70 sorting

70.1 sort2

```
sort two numbers
```

70.2 sort2d

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

71 special-functions

71.1 bessel_sphere

```
spherical Bessel function of the first kind
```

71.2 digamma_man

71.3 hankel_sphere

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

71.4 hermite

```
probabilistic's hermite polynomial by recurrence relation

input :
n : order
x : value

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

71.5 legendre_man

```
legendre polynomials
```

71.6 neumann_sphere

```
spherical Neumann function
Bessel function of the second kind
```

72 statistics

72.1 atan_s2

```
stadard deviation of the arcus tangens by means of taylor expansion
```

72.2 beta_mode_to_parameter

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

72.3 coefficient_of_determination

72.4 conditional_expectation_normal

72.5 correlation_confidence_pearson

```
confience intervals of the correlation coefficient
c.f. Fischer 1921
```

73 statistics/distributions

73.1 PDF

```
class for quasi-distributions from a set of sampling points
```

73.2 binorm_separation_coefficient

```
separation coefficient of a bimodal normal distribution
```

73.3 binormcdf

```
bio-modal gaussian distribution
```

73.4 binormfit

```
fit sum of to normal distribution to a histogram
```

73.5 binormpdf

73.6 edgeworth_cdf

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

73.7 edgeworth_pdf

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

73.8 logn_mode2param

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

73.9 logn_param2mode

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

73.10 lognpdf_

```
log normal distribution called by modes rather than parameters
```

73.11 pdfsample

```
pdf from sample distribution
Note: better use kernal density estimates
```

73.12 t2cdf

```
Hotelling's T-squared cumulative distribution
```

73.13 t2inv

```
inverse of Hotelling's T-squared cumulative distribution
```

74 statistics

74.1 example_standard_error_of_sample_quantiles

74.2 f_var_finite

```
reduction of variance when sampling from a finite population
without replacement
```

74.3 gamma_mode_to_parameter

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

74.4 gaussfit3

74.5 gaussfit_quantile

74.6 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
```

74.7 hodges_lehmann_dispersion

```
dispersion determined by the hodges lehman method
asymptotic efficiency of dispersion estimates:
standard deviation:   E(s - hat s)/s = 2/sqrt(2 n) ~ 0.707/sqrt(n)
               (100%)
hodges lehmann dispersion E(s-\hat s)/s = (pi/3)^2 /(sqrt(2 n)) ~
    0.775/sqrt(n) (91%)
```

```
mad                   E(s-\hat s)/s ~ 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
```

75 statistics/information-theory

75.1 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
```

75.2 bayesian_information_criterion

```
bayesian information criterion
```

76 statistics

76.1 kurtncdf

76.2 kurtnpdf

76.3 kurtosis_bias_corrected

```
bias corrected kurtosis
```

76.4 limit

```
limit a by lower and upper bound
```

76.5 logfactorial

```
approximate log of the factorial
```

76.6 loglogpdf

76.7 lognfit_quantile

76.8 logskewcdf

76.9 logskewpdf

77 statistics/logu

77.1 lambertw_numeric

```
lambert-w function
```

77.2 logtrialtcdf

```
pdf of a logarithmic triangular distribution
```

77.3 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))))
```

77.4 logtrialtmean

mean of the logarithmic triangular distribution

77.5 logtrialtpdf

density of the logarithmic triangular distribution

77.6 logtrialtrnd

77.7 logtricdf

cumulative distribution of the logarithmic triangular distribution

77.8 logtriinv

invere of the logarithmic triangular distribution

77.9 logtrimean

mean of the logarithmic triangular distribution

77.10 logtripdf

probability density of the logarithmic triangular distribution

77.11 logtrirnd

77.12 logucdf

probability density of the logarithmic uniform distribution

77.13 logucm

central moments of the log-uniform distribution

77.14 loguinv

inverse of the log-uniform distribution

77.15 logumean

mean of the log-uniform distribution

77.16 logupdf

pdf of the log uniform distribution

77.17 logurnd

`random numbers following a log-uniform distribution`

77.18 loguvar

`variance of the log-uniform distribution`

77.19 medlogu

`median of the log-uniform distribution`

77.20 test_logurnd

77.21 tricdf

`cumulative distribution of the log-triangular distribution`

77.22 triinv

`inverse of the triangular distribution`

77.23 trimedian

`median of the triangular distribution`

77.24 tripdf

`probability density of the triangular distribution`

77.25 trirnd

```
random numbers of the triangular distribution
```

78 statistics

78.1 maxnnormals

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

78.2 midrange

```
mid range of columns of X
```

78.3 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
```

78.4 mode_man

79 statistics/moment-statistics

79.1 autocorr_man3

```
autoccorrelation of the columns of X
```

79.2 autocorr_man4

```
autocorrelation for x if x is a vector, or indivvidually 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
```

79.3 autocorr_man5

```
autocorrellation of the columns of X
```

79.4 blockserr

```
estimate the standard error of potetially sequentilly 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
```

79.5 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 cii^2, the square seems to be missing
mu : nx1 mean vector
C  : nxn covariance matrix
k  : nx1 powers of variables in moments
```

79.6 corr_man

```
correlation of two vectors
```

79.7 cov_man

```
covariance matrix of two vectors
```

79.8 dof

```
mininum number of support points
for a polynomial of degree order in dim dimensions
```

79.9 edgeworth_quantile

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

79.10 effective_sample_size

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

79.11 f_correlation

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

79.12 f_finite

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

79.13 lmean

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

79.14 lmoment

```
l-moment of vector x
```

79.15 maskmean

```
mean of the masked values of X
```

79.16 masknanmean

79.17 mean1

```
mean of x
```

79.18 mean_man

```
mean and standard error of X
```

79.19 mse

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

79.20 nanautocorr_man1

```
autocorrelation of a vector with nan-values
```

79.21 nanautocorr_man2

```
autocorrelation of a vector with nan-values
```

79.22 nanautocorr_man4

```
compute autocorrelation for x if x is a vector, or indivvidually
    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
```

79.23 nancorr

```
(co)-correlation matrix when samples a NaN
```

79.24 nancumsum

```
cumulative sum, setting nan values to zero
```

79.25 nanlmean

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

79.26 nanr2

```
coefficient of determination when samples are invalid
```

79.27 nanrms

```
root mean square value when sample contains nan-values
```

79.28 nanrmse

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

79.29 nanserr

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

79.30 nanwmean

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

79.31 nanwstd

```
weighed standard deviation
```

79.32 nanwvar

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

s^2 = sum(w*(x-sum(wx)/sum(w))^2)/sum(w)
```

79.33 nanxcorr

79.34 pearson

```
pearson correlation coefficient
```

79.35 pearson_to_kendall

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

79.36 pool_samples

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

79.37 qmean

```
trimmed mean
```

79.38 range_mean

79.39 rmse_

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

79.40 serr

```
standard error of the mean of a set of uncorrelated samples
```

79.41 serr1

79.42 test_qskew

79.43 test_qstd_qskew_optimal_p

79.44 wautocorr

```
autocorrelation for x if x is a vector, or indivvidually 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
```

79.45 wcorr

```
correlation of two vectors when samples are weighted
```

79.46 wcov

```
covariance of two vectors when samples are weighted
```

79.47 wdof

```
effective degrees of freedom for weighted samples
```

79.48 wkurt

```
kurtosis with weighted samples
```

79.49 wmean

```
weighted mean

min_x sum w (x-mu)^2 => mu = sum(wx)/sum(w)

varargin can be dim
function [mu serr] = wmean(w,x)
```

79.50 wrms

```
weighted root mean square error
```

79.51 wserr

```
weighted root mean square error
```

79.52 wskew

```
skewness of a weighted set of samples
```

79.53 wstd

```
weighed standard deviation
```

79.54 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 = sum (w^2(x-sum(wx)^2))

s2_mu : error of mean, s2_mu : sd of prediction
```

80 statistics

80.1 nangeomean

80.2 nangeostd

```
geometric standard deviation ignoring nan-values
```

81 statistics/nonparametric-statistics

81.1 kernel1d

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

81.2 kernel2d

```
kernel density estimate in two dimensions
```

82 statistics

82.1 normmoment

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

82.2 normpdf2

```
pdf of the bivariate normal distribution
```

83 statistics/order-statistics

83.1 hodges_lehmann_location

```
hodges lehman location estimator

Asymptotic rms efficency of location estimte:
     mean:          1 s/sqrt(n)
     hodges lehman: sqrt(pi/3)*s ~ 1.0233 s/sqrt(n)
     median:        pi/2 s/sqrt(n) ~ 1.25 s / sqrt(n)
```

83.2 kendall

```
kendall correlation coefficient
```

83.3 kendall_to_pearson

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

c.f. Kruska, 1985
```

83.4 mad2sd

```
transform median absolute deviation to standard deviation
for normal distributed values
```

83.5 madcorr

```
proxy correlation by median absolute deviation
```

83.6 median2_holder

83.7 median_ci

```
median and its confidence intervals under assumption of normality
se_me = sqrt(1/2 pi) 1.25331 * sd/sqrt(n)
```

83.8 median_man

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

83.9 mediani

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

83.10 nanmadcorr

```
proxy correlation by median absolute deviation
```

83.11 nanwmedian

```
weighted median, skips nan-values
```

83.12 nanwquantile

```
weighted quantile, skips nan values
```

83.13 oja_median

```
two dimensional oja median
note: the multivariate median is not unique

oja 1983, for extension to multivariate function, see chaudhri
```

83.14 qkurtosis

```
kurosis 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
```

83.15 qmoments

```
moments estimated from quantiles
```

83.16 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
```

83.17 qskewq

```
skewness estimated by quantiles
```

83.18 qstdq

```
proxy standard deviation determined by quantiles
```

83.19 quantile1_optimisation

83.20 quantile2_breckling

```
qunatile regression
```

83.21 quantile2_chaudhuri

```
quantile regression
```

83.22 quantile2_projected

```
quantile in two dimensions
```

83.23 quantile2_projected2

```
spatial qunatile for chosen direction
```

83.24 quantile_envelope

83.25 quantile_regression_simple

```
simple quantile regression
```

83.26 ranking

```
ranking for spearman statistics
```

83.27 spatial_median

```
c.f. Oja 2008
is this the same as the oja simplex median (c.f. small 1990)?
```

83.28 spatial_quantile

```
spatial quantile
```

83.29 spatial_quantile2

```
spatial quantile
```

83.30 spatial_quantile3

```
spatial quantile
```

83.31 spatial_rank

```
unsigned rank
```

83.32 spatial_sign

```
spatial sign
```

83.33 spatial_signed_rank

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

83.34 spearman

```
spearman's product moment coefficient
```

83.35 spearman_rank

83.36 spearman_to_pearson

```
conversion of spearman rank to person product moment correlation
    coefficient
```

83.37 wmedian

```
weighted median
```

83.38 wquantile

```
weighted quantile
```

84 statistics

84.1 qstd

84.2 quantile_extrap

85 statistics/random-number-generation

85.1 laplacernd

```
random number of laplace distribution
```

85.2 randc

```
correlate to correlated standard normally distributed vectors
```

85.3 skewness2param

85.4 skewpdf_central_moments

85.5 skewrnd

```
random numbers of the skew normal distribution
```

85.6 skewrnd2

```
random numbers of the skew normal distribution
```

86 statistics

86.1 range

```
mid range
```

86.2 resample_with_replacement

87 statistics/resampling-statistics/@Jackknife

87.1 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 jacknife, 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 jacknife performs iferior to the d1
    jacknife
```

87.2 estimated_STATIC

```
jacknife 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
```

87.3 matrix1_STATIC

```
matrix of estimation for leaving out two samples at a time
```

87.4 matrix2

```
matrix of estimations for jacknive with two samples left out
```

88 statistics/resampling-statistics

88.1 block_jackknife

88.2 jackknife_moments

```
moments determined by the jacknife

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
```

88.3 moving_block_jackknife

```
blocked Jacknfife for autocorrelated data
sliding block, statistically more efficient but computationally
    expensive
note, number of blocks must be sufficiently large h ~ sqrt(n)? << n
```

88.4 randblockserr

```
standard error of sequentilly 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
```

88.5 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
```

89 statistics

89.1 scale_quantile_sd

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

89.2 sd_sample_quantiles

89.3 skewpdf

```
skew-normal distribution
c.f. Azzalini 1985
```

89.4 trimmed_mean

```
trimmed mean
```

89.5 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
```

89.6 ttest_man

```
two-sample t-test
unequal sample size
equal variance
```

89.7 ttest_paired

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

89.8 wgeomean

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

89.9 wgeovar

```
variance of the weighted geometric mean
```

89.10 wharmean

```
weighted harmonic mean
```

89.11 wharstd

89.12 wharvar

90 mathematics

```
mathematical functions of various kind
```

90.1 ternary_diagram

91 test/master

91.1 dat_test_lanczos_3d_k_20_n_40

91.2 poisson2d_blk

91.3 qr_implicit_givens_2

91.4 spectral_derivative_2d

91.5 test_2d_eigensolver_hydrogen

91.6 test_2d_refine

91.7 test_3d_eigensolver_hydrogen

91.8 test_FEM

91.9 test_Mesh_3d

91.10 test_arnoldi

91.11 test_arpackc

91.12 test_assemble

91.13 test_assembly_performance

91.14 test_bc_one_sided

91.15 test_compare_solvers

91.16 test_complete

91.17 test_convergence

91.18 test_convergence_b

91.19 test_df_2d

91.20 test_eig_algs

91.21 test_eig_inverse

91.22 test_eigs_lanczos

91.23 test_eigs_lanczos_1

91.24 test_eigs_lanczos_2

91.25 test_eigs_lanczos_performance

91.26 test_fdm

91.27 test_fdm_d_vargrid

91.28 test_fdm_spectral

91.29 test_fem

91.30 test_fem_1d

91.31 test_fem_1d_higher_order

91.32 test_fem_2d_adaptive

91.33 test_fem_2d_higher_order

91.34 test_fem_3d_higher_order

91.35 test_fem_3d_refine

91.36 test_fem_b

91.37 test_fem_derivative

91.38 test_fem_quadrature

91.39 test_final

91.40 test_fix_substitution

91.41 test_forward

91.42 test_get_sparse_arrays

91.43 test_harmonic_oscillator

91.44 test_high_order_fdm_periodic_bc

91.45 test_hydrogen_wf

91.46 test_ichol

91.47 test_interpolation

91.48 test_inverse_problem

91.49 test_it_vs_exact

91.50 test_jama

91.51 test_jd

91.52 test_jdqz

91.53 test_lanczos_2

91.54 test_lanczos_biorthogonal

91.55 test_laplacian

91.56 test_laplacian_non_uniform

91.57 test_laplacian_simple

91.58 test_mesh_2d_uniform

91.59 test_mesh_2d_uniform_2

91.60 test_mesh_circle

91.61 test_mesh_generation

91.62 test_mesh_interpolate

91.63 test_mg

91.64 test_minres_recycle

91.65 test_multigrid

91.66 test_nc

91.67 test_nonuniform_symmetric

91.68 test_pde

91.69 test_permutation

91.70 test_poison_fem

91.71 test_polar

91.72 test_potential

91.73 test_powers

91.74 test_precondition

91.75 test_project_rectangle

91.76 test_qr

91.77 test_quantum_well

91.78 test_radial_adaptive

91.79 test_radial_confinement

91.80 test_radial_fixes

91.81 test_refine_2d

91.82 test_refine_2d_b

91.83 test_refine_3d

91.84 test_refine_structural

91.85 test_regularisation

91.86 test_round_off

91.87 test_schrdinger_potentials

91.88 test_uniform_mesh

91.89 test_vargrid

92 test

92.1 test_gaussfit3

92.2 test_mtimes3x3

93 wavelet

93.1 contiuous_wavelet_transform

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

93.2 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
```

93.3 example_wavelets

93.4 phasewrap

```
wrap the phase to +/- pi
```

93.5 test_cwt_man

93.6 test_phasewrap

93.7 test_wavelet

93.8 test_wavelet2

93.9 test_wavelet_analysis

93.10 test_wavelet_reconstruct

93.11 test_wtc

93.12 wavelet

```
wavelet windows
```

93.13 wavelet_reconstruct

```
iverses wavelet transform for single frequency
(reconstruction of time series)
n : window lengths in multiples of filter period 1/f0
```

93.14 wavelet_transform

```
wavelet transform for single frequency
n : window lengths in multiples of filter period 1/f0
```

94 mathematics

```
mathematical functions of various kind
```

94.1 wrapphase