spheroidal_lib

0.1.0

Generated by Doxygen 1.9.1

1 spheroidal_cpp 1
1.1 Description . . . 1
1.2 Getting Started . . . 1
1.2.1 Dependencies . . . 1
1.3 Authors . . . 2
1.4 Version History . . . 2
1.5 Acknowledgments . . . 2

2 Namespace Index 3
2.1 Namespace List . . . 3

3 Hierarchical Index 5
3.1 Class Hierarchy . . . 5

4 Class Index 7
4.1 Class List . . . 7

5 File Index 9
5.1 File List . . . 9

6 Namespace Documentation 11
6.1 gsl Namespace Reference . . . 11
6.1.1 Enumeration Type Documentation . . . 13
6.1.1.1 legendre_norm . . . 13
6.1.2 Function Documentation . . . 13
6.1.2.1 arange() . . . 14
6.1.2.2 arrayfun() [1/8] . . . 14
6.1.2.3 arrayfun() [2/8] . . . 14
6.1.2.4 arrayfun() [3/8] . . . 14
6.1.2.5 arrayfun() [4/8] . . . 15
6.1.2.6 arrayfun() [5/8] . . . 15
6.1.2.7 arrayfun() [6/8] . . . 16
6.1.2.8 arrayfun() [7/8] . . . 16
6.1.2.9 arrayfun() [8/8] . . . 16
6.1.2.10 circshift() [1/2] . . . 16
6.1.2.11 circshift() [2/2] . . . 16
6.1.2.12 diag() . . . 17
6.1.2.13 eye() . . . 17
6.1.2.14 fft() [1/4] . . . 17
6.1.2.15 fft() [2/4] . . . 17
6.1.2.16 fft() [3/4] . . . 17
6.1.2.17 fft() [4/4] . . . 17
6.1.2.18 fit_linear() . . . 18
6.1.2.19 ifft() [1/4] . . . 18

6.1.2.20 ifft() [2/4] . . . 18
6.1.2.21 ifft() [3/4] . . . 18
6.1.2.22 ifft() [4/4] . . . 18
6.1.2.23 leggauss() [1/2] . . . 19
6.1.2.24 leggauss() [2/2] . . . 19
6.1.2.25 linspace() [1/2] . . . 19
6.1.2.26 linspace() [2/2] . . . 19
6.1.2.27 meshgrid() [1/2] . . . 20
6.1.2.28 meshgrid() [2/2] . . . 20
6.1.2.29 pow() [1/4] . . . 20
6.1.2.30 pow() [2/4] . . . 20
6.1.2.31 pow() [3/4] . . . 20
6.1.2.32 pow() [4/4] . . . 21
6.1.2.33 spherical_harmonic() [1/2] . . . 21
6.1.2.34 spherical_harmonic() [2/2] . . . 21
6.2 gsl::complex_literals Namespace Reference . . . 21
6.2.1 Function Documentation . . . 21
6.2.1.1 operator""_i() . . . 21

7 Class Documentation **23**

7.1 gsl::ccolumn_view Class Reference . . . 23
7.1.1 Constructor & Destructor Documentation . . . 24
7.1.1.1 ccolumn_view() . . . 24
7.1.2 Member Function Documentation . . . 24
7.1.2.1 operator=() . . . 25
7.2 gsl::cmatrix Class Reference . . . 25
7.2.1 Constructor & Destructor Documentation . . . 28
7.2.1.1 cmatrix() [1/9] . . . 28
7.2.1.2 cmatrix() [2/9] . . . 28
7.2.1.3 cmatrix() [3/9] . . . 28
7.2.1.4 cmatrix() [4/9] . . . 28
7.2.1.5 cmatrix() [5/9] . . . 28
7.2.1.6 cmatrix() [6/9] . . . 29
7.2.1.7 cmatrix() [7/9] . . . 29
7.2.1.8 cmatrix() [8/9] . . . 29
7.2.1.9 ∼cmatrix() . . . 29
7.2.1.10 cmatrix() [9/9] . . . 29
7.2.2 Member Function Documentation . . . 29
7.2.2.1 clear() . . . 29
7.2.2.2 column() . . . 30
7.2.2.3 conj() . . . 30
7.2.2.4 galloc() . . . 30

7.2.2.5 get() [1/2] . . . 30
7.2.2.6 get() [2/2] . . . 30
7.2.2.7 get_col() . . . 30
7.2.2.8 get_row() . . . 31
7.2.2.9 gfree() . . . 31
7.2.2.10 H() . . . 31
7.2.2.11 is_square() . . . 31
7.2.2.12 load_csv() . . . 31
7.2.2.13 ncols() . . . 31
7.2.2.14 nrows() . . . 31
7.2.2.15 operator()() [1/2] . . . 32
7.2.2.16 operator()() [2/2] . . . 32
7.2.2.17 operator∗=() [1/2] . . . 32
7.2.2.18 operator∗=() [2/2] . . . 32
7.2.2.19 operator+=() [1/2] . . . 32
7.2.2.20 operator+=() [2/2] . . . 32
7.2.2.21 operator-() . . . 33
7.2.2.22 operator-=() [1/2] . . . 33
7.2.2.23 operator-=() [2/2] . . . 33
7.2.2.24 operator/=() [1/2] . . . 33
7.2.2.25 operator/=() [2/2] . . . 33
7.2.2.26 operator=() [1/3] . . . 33
7.2.2.27 operator=() [2/3] . . . 33
7.2.2.28 operator=() [3/3] . . . 34
7.2.2.29 print() . . . 34
7.2.2.30 print_csv() . . . 34
7.2.2.31 reshape() . . . 34
7.2.2.32 resize() . . . 34
7.2.2.33 row() . . . 34
7.2.2.34 set() . . . 35
7.2.2.35 set_col() . . . 35
7.2.2.36 set_row() . . . 35
7.2.2.37 size() . . . 35
7.2.2.38 submatrix() . . . 35
7.2.2.39 T() . . . 35
7.2.2.40 view() . . . 36
7.2.3 Friends And Related Function Documentation . . . 36
7.2.3.1 operator"!= [1/3] . . . 36
7.2.3.2 operator"!= [2/3] . . . 36
7.2.3.3 operator"!= [3/3] . . . 36
7.2.3.4 operator∗ [1/10] . . . 36
7.2.3.5 operator∗ [2/10] . . . 36

7.2.3.6 operator* [3/10] . . . 37
7.2.3.7 operator* [4/10] . . . 37
7.2.3.8 operator* [5/10] . . . 37
7.2.3.9 operator* [6/10] . . . 37
7.2.3.10 operator* [7/10] . . . 37
7.2.3.11 operator* [8/10] . . . 37
7.2.3.12 operator* [9/10] . . . 38
7.2.3.13 operator* [10/10] . . . 38
7.2.3.14 operator+ [1/8] . . . 38
7.2.3.15 operator+ [2/8] . . . 38
7.2.3.16 operator+ [3/8] . . . 38
7.2.3.17 operator+ [4/8] . . . 38
7.2.3.18 operator+ [5/8] . . . 39
7.2.3.19 operator+ [6/8] . . . 39
7.2.3.20 operator+ [7/8] . . . 39
7.2.3.21 operator+ [8/8] . . . 39
7.2.3.22 operator- [1/8] . . . 39
7.2.3.23 operator- [2/8] . . . 39
7.2.3.24 operator- [3/8] . . . 40
7.2.3.25 operator- [4/8] . . . 40
7.2.3.26 operator- [5/8] . . . 40
7.2.3.27 operator- [6/8] . . . 40
7.2.3.28 operator- [7/8] . . . 40
7.2.3.29 operator- [8/8] . . . 40
7.2.3.30 operator/ [1/8] . . . 41
7.2.3.31 operator/ [2/8] . . . 41
7.2.3.32 operator/ [3/8] . . . 41
7.2.3.33 operator/ [4/8] . . . 41
7.2.3.34 operator/ [5/8] . . . 41
7.2.3.35 operator/ [6/8] . . . 41
7.2.3.36 operator/ [7/8] . . . 42
7.2.3.37 operator/ [8/8] . . . 42
7.2.3.38 operator== [1/3] . . . 42
7.2.3.39 operator== [2/3] . . . 42
7.2.3.40 operator== [3/3] . . . 42
7.2.4 Member Data Documentation . . . 42
7.2.4.1 gmat . . . 42
7.3 gsl::cmatrix_view Class Reference . . . 43
7.3.1 Constructor & Destructor Documentation . . . 43
7.3.1.1 cmatrix_view() . . . 44
7.3.1.2 ~cmatrix_view() . . . 44
7.3.2 Member Function Documentation . . . 44

7.3.2.1 clear() . . . 44
7.3.2.2 operator=() . . . 44
7.3.2.3 resize() . . . 44
7.4 gsl::column_view Class Reference . . . 45
7.4.1 Detailed Description . . . 46
7.4.2 Constructor & Destructor Documentation . . . 46
7.4.2.1 column_view() . . . 46
7.4.3 Member Function Documentation . . . 46
7.4.3.1 operator=() . . . 46
7.5 gsl::complex Class Reference . . . 47
7.5.1 Detailed Description . . . 48
7.5.2 Constructor & Destructor Documentation . . . 48
7.5.2.1 complex() [1/4] . . . 48
7.5.2.2 complex() [2/4] . . . 49
7.5.2.3 complex() [3/4] . . . 49
7.5.2.4 complex() [4/4] . . . 49
7.5.3 Member Function Documentation . . . 49
7.5.3.1 abs() . . . 49
7.5.3.2 abs2() . . . 49
7.5.3.3 arg() . . . 49
7.5.3.4 imag() . . . 49
7.5.3.5 operator*=() [1/2] . . . 50
7.5.3.6 operator*=() [2/2] . . . 50
7.5.3.7 operator+=() [1/2] . . . 50
7.5.3.8 operator+=() [2/2] . . . 50
7.5.3.9 operator-() . . . 50
7.5.3.10 operator-=() [1/2] . . . 50
7.5.3.11 operator-=() [2/2] . . . 50
7.5.3.12 operator/=() [1/2] . . . 51
7.5.3.13 operator/=() [2/2] . . . 51
7.5.3.14 operator=() . . . 51
7.5.3.15 print() . . . 51
7.5.3.16 real() . . . 51
7.5.3.17 set() [1/2] . . . 51
7.5.3.18 set() [2/2] . . . 51
7.5.4 Friends And Related Function Documentation . . . 52
7.5.4.1 complex_ref . . . 52
7.5.4.2 operator"!= [1/3] . . . 52
7.5.4.3 operator"!= [2/3] . . . 52
7.5.4.4 operator"!= [3/3] . . . 52
7.5.4.5 operator* [1/3] . . . 52
7.5.4.6 operator* [2/3] . . . 52

7.5.4.7 operator* [3/3] . . . 53
7.5.4.8 operator+ [1/3] . . . 53
7.5.4.9 operator+ [2/3] . . . 53
7.5.4.10 operator+ [3/3] . . . 53
7.5.4.11 operator- [1/3] . . . 53
7.5.4.12 operator- [2/3] . . . 53
7.5.4.13 operator- [3/3] . . . 54
7.5.4.14 operator/ [1/3] . . . 54
7.5.4.15 operator/ [2/3] . . . 54
7.5.4.16 operator/ [3/3] . . . 54
7.5.4.17 operator== [1/3] . . . 54
7.5.4.18 operator== [2/3] . . . 54
7.5.4.19 operator== [3/3] . . . 55
7.6 gsl::complex_ref Class Reference . . . 55
7.6.1 Detailed Description . . . 56
7.6.2 Constructor & Destructor Documentation . . . 56
7.6.2.1 complex_ref() [1/4] . . . 56
7.6.2.2 complex_ref() [2/4] . . . 57
7.6.2.3 complex_ref() [3/4] . . . 57
7.6.2.4 complex_ref() [4/4] . . . 57
7.6.3 Member Function Documentation . . . 57
7.6.3.1 imag() . . . 57
7.6.3.2 operator complex() . . . 57
7.6.3.3 operator*() . . . 57
7.6.3.4 operator*=() [1/2] . . . 58
7.6.3.5 operator*=() [2/2] . . . 58
7.6.3.6 operator+=() [1/2] . . . 58
7.6.3.7 operator+=() [2/2] . . . 58
7.6.3.8 operator-=() [1/2] . . . 58
7.6.3.9 operator-=() [2/2] . . . 58
7.6.3.10 operator/=() [1/2] . . . 58
7.6.3.11 operator/=() [2/2] . . . 59
7.6.3.12 operator=() [1/2] . . . 59
7.6.3.13 operator=() [2/2] . . . 59
7.6.3.14 real() . . . 59
7.6.4 Friends And Related Function Documentation . . . 59
7.6.4.1 complex . . . 59
7.6.4.2 operator"!= [1/3] . . . 60
7.6.4.3 operator"!= [2/3] . . . 60
7.6.4.4 operator"!= [3/3] . . . 60
7.6.4.5 operator* [1/3] . . . 60
7.6.4.6 operator* [2/3] . . . 60

7.6.4.7 operator* [3/3] . . . 60
7.6.4.8 operator+ [1/3] . . . 61
7.6.4.9 operator+ [2/3] . . . 61
7.6.4.10 operator+ [3/3] . . . 61
7.6.4.11 operator- [1/3] . . . 61
7.6.4.12 operator- [2/3] . . . 61
7.6.4.13 operator- [3/3] . . . 61
7.6.4.14 operator/ [1/3] . . . 62
7.6.4.15 operator/ [2/3] . . . 62
7.6.4.16 operator/ [3/3] . . . 62
7.6.4.17 operator== [1/3] . . . 62
7.6.4.18 operator== [2/3] . . . 62
7.6.4.19 operator== [3/3] . . . 62
7.6.5 Member Data Documentation . . . 63
7.6.5.1 dat . . . 63
7.7 gsl::crow_view Class Reference . . . 63
7.7.1 Detailed Description . . . 64
7.7.2 Constructor & Destructor Documentation . . . 64
7.7.2.1 crow_view() . . . 64
7.7.3 Member Function Documentation . . . 65
7.7.3.1 operator=() . . . 65
7.7.3.2 reshape() . . . 65
7.8 gsl::cvector Class Reference . . . 65
7.8.1 Detailed Description . . . 67
7.8.2 Constructor & Destructor Documentation . . . 68
7.8.2.1 cvector() [1/7] . . . 68
7.8.2.2 cvector() [2/7] . . . 68
7.8.2.3 cvector() [3/7] . . . 68
7.8.2.4 cvector() [4/7] . . . 68
7.8.2.5 cvector() [5/7] . . . 68
7.8.2.6 cvector() [6/7] . . . 68
7.8.2.7 ~cvector() . . . 69
7.8.2.8 cvector() [7/7] . . . 69
7.8.3 Member Function Documentation . . . 69
7.8.3.1 clear() . . . 69
7.8.3.2 galloc() . . . 69
7.8.3.3 get() [1/2] . . . 69
7.8.3.4 get() [2/2] . . . 69
7.8.3.5 gfree() . . . 70
7.8.3.6 norm() . . . 70
7.8.3.7 operator()() [1/2] . . . 70
7.8.3.8 operator()() [2/2] . . . 70

7.8.3.9 operator*=() [1/2] . . . 70
7.8.3.10 operator*=() [2/2] . . . 70
7.8.3.11 operator+=() [1/2] . . . 71
7.8.3.12 operator+=() [2/2] . . . 71
7.8.3.13 operator-() . . . 71
7.8.3.14 operator-=() [1/2] . . . 71
7.8.3.15 operator-=() [2/2] . . . 71
7.8.3.16 operator/=() [1/2] . . . 71
7.8.3.17 operator/=() [2/2] . . . 71
7.8.3.18 operator=() [1/3] . . . 72
7.8.3.19 operator=() [2/3] . . . 72
7.8.3.20 operator=() [3/3] . . . 72
7.8.3.21 print() . . . 72
7.8.3.22 resize() . . . 72
7.8.3.23 set() . . . 72
7.8.3.24 size() . . . 73
7.8.3.25 subvector() . . . 73
7.8.3.26 view() . . . 73
7.8.4 Friends And Related Function Documentation . . . 73
7.8.4.1 operator"!= [1/3] . . . 73
7.8.4.2 operator"!= [2/3] . . . 73
7.8.4.3 operator"!= [3/3] . . . 73
7.8.4.4 operator* [1/9] . . . 74
7.8.4.5 operator* [2/9] . . . 74
7.8.4.6 operator* [3/9] . . . 74
7.8.4.7 operator* [4/9] . . . 74
7.8.4.8 operator* [5/9] . . . 74
7.8.4.9 operator* [6/9] . . . 74
7.8.4.10 operator* [7/9] . . . 75
7.8.4.11 operator* [8/9] . . . 75
7.8.4.12 operator* [9/9] . . . 75
7.8.4.13 operator+ [1/8] . . . 75
7.8.4.14 operator+ [2/8] . . . 75
7.8.4.15 operator+ [3/8] . . . 75
7.8.4.16 operator+ [4/8] . . . 76
7.8.4.17 operator+ [5/8] . . . 76
7.8.4.18 operator+ [6/8] . . . 76
7.8.4.19 operator+ [7/8] . . . 76
7.8.4.20 operator+ [8/8] . . . 76
7.8.4.21 operator- [1/8] . . . 76
7.8.4.22 operator- [2/8] . . . 77
7.8.4.23 operator- [3/8] . . . 77

7.8.4.24 operator- [4/8] . . . 77
7.8.4.25 operator- [5/8] . . . 77
7.8.4.26 operator- [6/8] . . . 77
7.8.4.27 operator- [7/8] . . . 77
7.8.4.28 operator- [8/8] . . . 78
7.8.4.29 operator/ [1/8] . . . 78
7.8.4.30 operator/ [2/8] . . . 78
7.8.4.31 operator/ [3/8] . . . 78
7.8.4.32 operator/ [4/8] . . . 78
7.8.4.33 operator/ [5/8] . . . 78
7.8.4.34 operator/ [6/8] . . . 79
7.8.4.35 operator/ [7/8] . . . 79
7.8.4.36 operator/ [8/8] . . . 79
7.8.4.37 operator== [1/3] . . . 79
7.8.4.38 operator== [2/3] . . . 79
7.8.4.39 operator== [3/3] . . . 79
7.8.5 Member Data Documentation . . . 80
7.8.5.1 gvec . . . 80
7.9 gsl::cvector_view Class Reference . . . 80
7.9.1 Constructor & Destructor Documentation . . . 81
7.9.1.1 cvector_view() . . . 81
7.9.1.2 ∼cvector_view() . . . 81
7.9.2 Member Function Documentation . . . 81
7.9.2.1 clear() . . . 81
7.9.2.2 operator=() . . . 81
7.9.2.3 resize() . . . 82
7.10 gsl::matrix Class Reference . . . 82
7.10.1 Detailed Description . . . 84
7.10.2 Constructor & Destructor Documentation . . . 84
7.10.2.1 matrix() [1/8] . . . 85
7.10.2.2 matrix() [2/8] . . . 85
7.10.2.3 matrix() [3/8] . . . 85
7.10.2.4 matrix() [4/8] . . . 85
7.10.2.5 matrix() [5/8] . . . 85
7.10.2.6 matrix() [6/8] . . . 86
7.10.2.7 matrix() [7/8] . . . 86
7.10.2.8 ∼matrix() . . . 86
7.10.2.9 matrix() [8/8] . . . 86
7.10.3 Member Function Documentation . . . 86
7.10.3.1 clear() . . . 86
7.10.3.2 column() . . . 86
7.10.3.3 galloc() . . . 87

7.10.3.4 get() [1/2] . . . 87
7.10.3.5 get() [2/2] . . . 87
7.10.3.6 get_col() . . . 87
7.10.3.7 get_row() . . . 87
7.10.3.8 gfree() . . . 87
7.10.3.9 is_square() . . . 88
7.10.3.10 load_csv() . . . 88
7.10.3.11 ncols() . . . 88
7.10.3.12 nrows() . . . 88
7.10.3.13 operator()() [1/2] . . . 88
7.10.3.14 operator()() [2/2] . . . 88
7.10.3.15 operator*=() . . . 88
7.10.3.16 operator+=() . . . 89
7.10.3.17 operator-() . . . 89
7.10.3.18 operator-=() . . . 89
7.10.3.19 operator/=() . . . 89
7.10.3.20 operator=() [1/2] . . . 89
7.10.3.21 operator=() [2/2] . . . 89
7.10.3.22 print() . . . 89
7.10.3.23 print_csv() . . . 90
7.10.3.24 reshape() . . . 90
7.10.3.25 resize() . . . 90
7.10.3.26 row() . . . 90
7.10.3.27 set() . . . 90
7.10.3.28 set_col() . . . 90
7.10.3.29 set_row() . . . 91
7.10.3.30 size() . . . 91
7.10.3.31 submatrix() . . . 91
7.10.3.32 T() . . . 91
7.10.3.33 view() . . . 91
7.10.4 Friends And Related Function Documentation . . . 91
7.10.4.1 operator"!= [1/3] . . . 91
7.10.4.2 operator"!= [2/3] . . . 92
7.10.4.3 operator"!= [3/3] . . . 92
7.10.4.4 operator* [1/7] . . . 92
7.10.4.5 operator* [2/7] . . . 92
7.10.4.6 operator* [3/7] . . . 92
7.10.4.7 operator* [4/7] . . . 92
7.10.4.8 operator* [5/7] . . . 93
7.10.4.9 operator* [6/7] . . . 93
7.10.4.10 operator* [7/7] . . . 93
7.10.4.11 operator+ [1/8] . . . 93

7.10.4.12 operator+ [2/8] . . . 93
7.10.4.13 operator+ [3/8] . . . 93
7.10.4.14 operator+ [4/8] . . . 94
7.10.4.15 operator+ [5/8] . . . 94
7.10.4.16 operator+ [6/8] . . . 94
7.10.4.17 operator+ [7/8] . . . 94
7.10.4.18 operator+ [8/8] . . . 94
7.10.4.19 operator- [1/8] . . . 94
7.10.4.20 operator- [2/8] . . . 95
7.10.4.21 operator- [3/8] . . . 95
7.10.4.22 operator- [4/8] . . . 95
7.10.4.23 operator- [5/8] . . . 95
7.10.4.24 operator- [6/8] . . . 95
7.10.4.25 operator- [7/8] . . . 95
7.10.4.26 operator- [8/8] . . . 96
7.10.4.27 operator/ [1/6] . . . 96
7.10.4.28 operator/ [2/6] . . . 96
7.10.4.29 operator/ [3/6] . . . 96
7.10.4.30 operator/ [4/6] . . . 96
7.10.4.31 operator/ [5/6] . . . 96
7.10.4.32 operator/ [6/6] . . . 97
7.10.4.33 operator== [1/3] . . . 97
7.10.4.34 operator== [2/3] . . . 97
7.10.4.35 operator== [3/3] . . . 97
7.10.5 Member Data Documentation . . . 97
7.10.5.1 gmat . . . 97
7.11 gsl::matrix_view Class Reference . . . 98
7.11.1 Constructor & Destructor Documentation . . . 98
7.11.1.1 matrix_view() . . . 99
7.11.1.2 ~matrix_view() . . . 99
7.11.2 Member Function Documentation . . . 99
7.11.2.1 clear() . . . 99
7.11.2.2 operator=() . . . 99
7.11.2.3 resize() . . . 99
7.12 gsl::row_view Class Reference . . . 100
7.12.1 Detailed Description . . . 101
7.12.2 Constructor & Destructor Documentation . . . 101
7.12.2.1 row_view() . . . 101
7.12.3 Member Function Documentation . . . 101
7.12.3.1 operator=() . . . 101
7.12.3.2 reshape() . . . 101
7.13 gsl::vector Class Reference . . . 102

7.13.1 Detailed Description . . . 104
7.13.2 Constructor & Destructor Documentation . . . 104
7.13.2.1 vector() [1/5] . . . 104
7.13.2.2 vector() [2/5] . . . 104
7.13.2.3 vector() [3/5] . . . 104
7.13.2.4 vector() [4/5] . . . 104
7.13.2.5 ∼vector() . . . 105
7.13.2.6 vector() [5/5] . . . 105
7.13.3 Member Function Documentation . . . 105
7.13.3.1 clear() . . . 105
7.13.3.2 galloc() . . . 105
7.13.3.3 get() [1/2] . . . 105
7.13.3.4 get() [2/2] . . . 105
7.13.3.5 gfree() . . . 106
7.13.3.6 norm() . . . 106
7.13.3.7 operator()() [1/2] . . . 106
7.13.3.8 operator()() [2/2] . . . 106
7.13.3.9 operator∗=() . . . 106
7.13.3.10 operator+=() . . . 106
7.13.3.11 operator-() . . . 106
7.13.3.12 operator-=() . . . 107
7.13.3.13 operator/=() . . . 107
7.13.3.14 operator=() [1/2] . . . 107
7.13.3.15 operator=() [2/2] . . . 107
7.13.3.16 print() . . . 107
7.13.3.17 resize() . . . 107
7.13.3.18 set() . . . 108
7.13.3.19 size() . . . 108
7.13.3.20 subvector() . . . 108
7.13.3.21 view() . . . 108
7.13.4 Friends And Related Function Documentation . . . 108
7.13.4.1 operator"!= [1/3] . . . 108
7.13.4.2 operator"!= [2/3] . . . 108
7.13.4.3 operator"!= [3/3] . . . 109
7.13.4.4 operator∗ [1/7] . . . 109
7.13.4.5 operator∗ [2/7] . . . 109
7.13.4.6 operator∗ [3/7] . . . 109
7.13.4.7 operator∗ [4/7] . . . 109
7.13.4.8 operator∗ [5/7] . . . 109
7.13.4.9 operator∗ [6/7] . . . 110
7.13.4.10 operator∗ [7/7] . . . 110
7.13.4.11 operator+ [1/8] . . . 110

7.13.4.12 operator+ [2/8] . . . 110
7.13.4.13 operator+ [3/8] . . . 110
7.13.4.14 operator+ [4/8] . . . 110
7.13.4.15 operator+ [5/8] . . . 111
7.13.4.16 operator+ [6/8] . . . 111
7.13.4.17 operator+ [7/8] . . . 111
7.13.4.18 operator+ [8/8] . . . 111
7.13.4.19 operator- [1/8] . . . 111
7.13.4.20 operator- [2/8] . . . 111
7.13.4.21 operator- [3/8] . . . 112
7.13.4.22 operator- [4/8] . . . 112
7.13.4.23 operator- [5/8] . . . 112
7.13.4.24 operator- [6/8] . . . 112
7.13.4.25 operator- [7/8] . . . 112
7.13.4.26 operator- [8/8] . . . 112
7.13.4.27 operator/ [1/6] . . . 113
7.13.4.28 operator/ [2/6] . . . 113
7.13.4.29 operator/ [3/6] . . . 113
7.13.4.30 operator/ [4/6] . . . 113
7.13.4.31 operator/ [5/6] . . . 113
7.13.4.32 operator/ [6/6] . . . 113
7.13.4.33 operator== [1/3] . . . 114
7.13.4.34 operator== [2/3] . . . 114
7.13.4.35 operator== [3/3] . . . 114
7.13.5 Member Data Documentation . . . 114
7.13.5.1 gvec . . . 114
7.14 gsl::vector_view Class Reference . . . 115
7.14.1 Constructor & Destructor Documentation . . . 116
7.14.1.1 vector_view() . . . 116
7.14.1.2 ∼vector_view() . . . 116
7.14.2 Member Function Documentation . . . 116
7.14.2.1 clear() . . . 116
7.14.2.2 operator=() . . . 116
7.14.2.3 resize() . . . 116

8 File Documentation **117**

8.1 grid_functions.h File Reference . . . 117
8.1.1 Function Documentation . . . 118
8.1.1.1 gl_grid() . . . 118
8.1.1.2 spharm_grid_size_ord() . . . 118
8.1.1.3 spharm_grid_size_tot() . . . 118
8.2 legendre_otc.h File Reference . . . 119

8.2.1 Function Documentation . . . 119
8.2.1.1 cont_frac() . . . 119
8.2.1.2 Dlegendre_otc() [1/2] . . . 119
8.2.1.3 Dlegendre_otc() [2/2] . . . 120
8.2.1.4 geti() . . . 120
8.2.1.5 legendre_otc() [1/2] . . . 120
8.2.1.6 legendre_otc() [2/2] . . . 120
8.3 spheroidal_coordinate_functions.h File Reference . . . 120
8.3.1 Function Documentation . . . 121
8.3.1.1 cart_to_spheroidal() . . . 121
8.3.1.2 spheroidal_to_cart() . . . 121
8.4 spheroidal_double_layer.h File Reference . . . 122
8.4.1 Function Documentation . . . 122
8.4.1.1 DLspectrum() . . . 123
8.4.1.2 solid_harmonic() . . . 123
8.4.1.3 spheroidal_double_layer() [1/2] . . . 123
8.4.1.4 spheroidal_double_layer() [2/2] . . . 123
8.4.1.5 Ynm_matrix() . . . 123
8.5 spheroidal_harmonic_transforms.h File Reference . . . 124
8.5.1 Function Documentation . . . 124
8.5.1.1 get_legendre_matrix() . . . 124
8.5.1.2 get_legendre_matrix_inv() . . . 124
8.5.1.3 spheroidal_analysis() . . . 125
8.5.1.4 spheroidal_snythesis() . . . 125
8.6 /home/jspainhour/spheroidal_cpp/include/yawg/cmatrix.h File Reference . . . 125
8.7 /home/jspainhour/spheroidal_cpp/include/yawg/complex.h File Reference . . . 126
8.8 /home/jspainhour/spheroidal_cpp/include/yawg/core.h File Reference . . . 127
8.9 /home/jspainhour/spheroidal_cpp/include/yawg/cvector.h File Reference . . . 127
8.10 /home/jspainhour/spheroidal_cpp/include/yawg/fft.h File Reference . . . 128
8.11 /home/jspainhour/spheroidal_cpp/include/yawg/legendre.h File Reference . . . 130
8.12 /home/jspainhour/spheroidal_cpp/include/yawg/lls.h File Reference . . . 131
8.13 /home/jspainhour/spheroidal_cpp/include/yawg/matrix.h File Reference . . . 132
8.14 /home/jspainhour/spheroidal_cpp/include/yawg/utils.hpp File Reference . . . 133
8.15 /home/jspainhour/spheroidal_cpp/include/yawg/vector.h File Reference . . . 134
8.16 /home/jspainhour/spheroidal_cpp/README.md File Reference . . . 135
8.17 /home/jspainhour/spheroidal_cpp/tests/spheroidal/verification_tests.cpp File Reference . . . 135
8.17.1 Macro Definition Documentation . . . 136
8.17.1.1 CATCH_CONFIG_MAIN . . . 136
8.17.1.2 HW6_PLOTS . . . 136
8.17.2 Function Documentation . . . 137
8.17.2.1 PtChargePotential() . . . 137
8.17.2.2 TEST_CASE() . . . 137

8.17.2.3 test_density() . . . 137
8.18 /home/jspainhour/spheroidal_cpp/tests/yawg/test_fft.cpp File Reference . . . 137
8.18.1 Macro Definition Documentation . . . 138
8.18.1.1 CATCH_CONFIG_MAIN . . . 138
8.18.2 Function Documentation . . . 138
8.18.2.1 TEST_CASE() [1/5] . . . 138
8.18.2.2 TEST_CASE() [2/5] . . . 138
8.18.2.3 TEST_CASE() [3/5] . . . 138
8.18.2.4 TEST_CASE() [4/5] . . . 139
8.18.2.5 TEST_CASE() [5/5] . . . 139
8.19 /home/jspainhour/spheroidal_cpp/tests/yawg/test_utils.cpp File Reference . . . 139
8.19.1 Macro Definition Documentation . . . 140
8.19.1.1 CATCH_CONFIG_MAIN . . . 140
8.19.2 Function Documentation . . . 140
8.19.2.1 TEST_CASE() [1/9] . . . 140
8.19.2.2 TEST_CASE() [2/9] . . . 140
8.19.2.3 TEST_CASE() [3/9] . . . 140
8.19.2.4 TEST_CASE() [4/9] . . . 140
8.19.2.5 TEST_CASE() [5/9] . . . 141
8.19.2.6 TEST_CASE() [6/9] . . . 141
8.19.2.7 TEST_CASE() [7/9] . . . 141
8.19.2.8 TEST_CASE() [8/9] . . . 141
8.19.2.9 TEST_CASE() [9/9] . . . 141
8.20 /home/jspainhour/spheroidal_cpp/tests/yawg/test_views.cpp File Reference . . . 142
8.20.1 Macro Definition Documentation . . . 142
8.20.1.1 CATCH_CONFIG_MAIN . . . 142
8.20.2 Function Documentation . . . 142
8.20.2.1 TEST_CASE() [1/6] . . . 143
8.20.2.2 TEST_CASE() [2/6] . . . 143
8.20.2.3 TEST_CASE() [3/6] . . . 143
8.20.2.4 TEST_CASE() [4/6] . . . 143
8.20.2.5 TEST_CASE() [5/6] . . . 143
8.20.2.6 TEST_CASE() [6/6] . . . 143
8.21 /home/jspainhour/spheroidal_cpp/tests/yawg/test_wrapper.cpp File Reference . . . 144
8.21.1 Macro Definition Documentation . . . 145
8.21.1.1 CATCH_CONFIG_MAIN . . . 145
8.21.2 Function Documentation . . . 145
8.21.2.1 TEST_CASE() [1/24] . . . 145
8.21.2.2 TEST_CASE() [2/24] . . . 145
8.21.2.3 TEST_CASE() [3/24] . . . 145
8.21.2.4 TEST_CASE() [4/24] . . . 145
8.21.2.5 TEST_CASE() [5/24] . . . 146

8.21.2.6 TEST_CASE() [6/24] . . . 146
8.21.2.7 TEST_CASE() [7/24] . . . 146
8.21.2.8 TEST_CASE() [8/24] . . . 146
8.21.2.9 TEST_CASE() [9/24] . . . 146
8.21.2.10 TEST_CASE() [10/24] . . . 146
8.21.2.11 TEST_CASE() [11/24] . . . 147
8.21.2.12 TEST_CASE() [12/24] . . . 147
8.21.2.13 TEST_CASE() [13/24] . . . 147
8.21.2.14 TEST_CASE() [14/24] . . . 147
8.21.2.15 TEST_CASE() [15/24] . . . 147
8.21.2.16 TEST_CASE() [16/24] . . . 147
8.21.2.17 TEST_CASE() [17/24] . . . 148
8.21.2.18 TEST_CASE() [18/24] . . . 148
8.21.2.19 TEST_CASE() [19/24] . . . 148
8.21.2.20 TEST_CASE() [20/24] . . . 148
8.21.2.21 TEST_CASE() [21/24] . . . 148
8.21.2.22 TEST_CASE() [22/24] . . . 148
8.21.2.23 TEST_CASE() [23/24] . . . 149
8.21.2.24 TEST_CASE() [24/24] . . . 149
8.22 /home/jspainhour/spheroidal_cpp/tests/yawg/test_wrapper_math.cpp File Reference . . . 149
8.22.1 Macro Definition Documentation . . . 150
8.22.1.1 CATCH_CONFIG_MAIN . . . 150
8.22.2 Function Documentation . . . 150
8.22.2.1 TEST_CASE() [1/7] . . . 150
8.22.2.2 TEST_CASE() [2/7] . . . 150
8.22.2.3 TEST_CASE() [3/7] . . . 150
8.22.2.4 TEST_CASE() [4/7] . . . 150
8.22.2.5 TEST_CASE() [5/7] . . . 151
8.22.2.6 TEST_CASE() [6/7] . . . 151
8.22.2.7 TEST_CASE() [7/7] . . . 151

Chapter 1

spheroidal_cpp

1.1 Description

`spheroidal` is a library for constructing and manipulating spheroidal harmonics, evaluating boundary integral operators, and other operations necessary for solving boundary integral equations on spheroids. Additionally, this repository contains example files to demonstrate usage and test files to demonstrate correctness.

Additionally, this repository contains Yet Another Wrapper for GSL, or `yawg`. This library contains a lightweight interface for the GNU Scientific Library, or GSL. Its intentions are to simplify usage of these functions, such as with a `gsl::vector` class that automatically handles memory allocation and pointer management.

The full documentation is built with `doxygen`, and can be constructed by performing the `make docs` command. A .pdf prototype of this documentation can be found in the `docs/` subdirectory.

1.2 Getting Started

Both the `spheroidal` and `yawg` libraries are built using CMake. To build from the command line, run

```
mkdir build
cd build
cmake ..
make (<specific target>)
```

An extensive library of testing functions and assertions is contained in the `tests` subfolder. These tests can also be built and executed using CMake.

1.2.1 Dependencies

Requires the following prerequisitres, along with the version used during testing:

- C++11
- CMake v3.22.1
- GNU Scientific Library v2.7.1
- Doxygen v1.9.1 (optional)
- Catch2 v3.0.1

1.3 Authors

[Jacob Spainhour](@jcs15c)

[Leo Crowder](@lcrowder)

1.4 Version History

- 0.1
  - Initial Release (Currently in developmet)

1.5 Acknowledgments

We would like to acknowledge the existence of and demonstrate gratitude towards other GSL C++ wrappers, and hope that ours is comparable in utility.

The following are existing C++ gsl wrapper classes we have found.

- GSLwrap: https://gslwrap.sourceforge.net/
- ccgsl: https://ccgsl.sourceforge.net/
- GSL-lib: https://github.com/johanjoensson/GSL-lib
- ROOT: https://root.cern/root/html606/index.html

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

gsl 11
gsl::complex_literals 21

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

gsl::cmatrix . . . 25
  gsl::cmatrix_view . . . 43
gsl::complex_ref . . . 55
gsl::cvector . . . 65
  gsl::cvector_view . . . 80
    gsl::ccolumn_view . . . 23
    gsl::crow_view . . . 63
gsl_complex
  gsl::complex . . . 47
gsl::matrix . . . 82
  gsl::matrix_view . . . 98
gsl::vector . . . 102
  gsl::vector_view . . . 115
    gsl::column_view . . . 45
    gsl::row_view . . . 100

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

gsl::ccolumn_view . . . 23
gsl::cmatrix . . . 25
gsl::cmatrix_view . . . 43
gsl::column_view
A subclass of vector_view for non-stride-1 vectors . . . 45
gsl::complex
Wrapper class for gsl_complex structs . . . 47
gsl::complex_ref
Stores a refernce to a gsl::complex object . . . 55
gsl::crow_view
A subclass of cvector_view for stride-1 cvectors . . . 63
gsl::cvector
A wrapper class for gsl_vector_complex . . . 65
gsl::cvector_view . . . 80
gsl::matrix
A wrapper class for gsl_matrix . . . 82
gsl::matrix_view . . . 98
gsl::row_view
A subclass of cvector_view for stride-1 cvectors . . . 100
gsl::vector
A wrapper class for gsl_vector . . . 102
gsl::vector_view . . . 115

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

grid_functions.h . . . 117
legendre_otc.h . . . 119
spheroidal_coordinate_functions.h . . . 120
spheroidal_double_layer.h . . . 122
spheroidal_harmonic_transforms.h . . . 124
/home/jspainhour/spheroidal_cpp/include/yawg/cmatrix.h . . . 125
/home/jspainhour/spheroidal_cpp/include/yawg/complex.h . . . 126
/home/jspainhour/spheroidal_cpp/include/yawg/core.h . . . 127
/home/jspainhour/spheroidal_cpp/include/yawg/cvector.h . . . 127
/home/jspainhour/spheroidal_cpp/include/yawg/fft.h . . . 128
/home/jspainhour/spheroidal_cpp/include/yawg/legendre.h . . . 130
/home/jspainhour/spheroidal_cpp/include/yawg/lls.h . . . 131
/home/jspainhour/spheroidal_cpp/include/yawg/matrix.h . . . 132
/home/jspainhour/spheroidal_cpp/include/yawg/utils.hpp . . . 133
/home/jspainhour/spheroidal_cpp/include/yawg/vector.h . . . 134
/home/jspainhour/spheroidal_cpp/src_spheroidal/grid_functions.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_spheroidal/legendre_otc.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_spheroidal/spheroidal_coordinate_functions.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_spheroidal/spheroidal_double_layer.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_spheroidal/spheroidal_harmonic_transforms.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/cmatrix.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/complex.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/cvector.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/fft.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/legendre.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/lls.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/matrix.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/utils.cpp . . . ??
/home/jspainhour/spheroidal_cpp/src_yawg/vector.cpp . . . ??
/home/jspainhour/spheroidal_cpp/tests/spheroidal/verification_tests.cpp . . . 135
/home/jspainhour/spheroidal_cpp/tests/yawg/test_fft.cpp . . . 137
/home/jspainhour/spheroidal_cpp/tests/yawg/test_utils.cpp . . . 139
/home/jspainhour/spheroidal_cpp/tests/yawg/test_views.cpp . . . 142
/home/jspainhour/spheroidal_cpp/tests/yawg/test_wrapper.cpp . . . 144
/home/jspainhour/spheroidal_cpp/tests/yawg/test_wrapper_math.cpp . . . 149

Chapter 6

Namespace Documentation

6.1 gsl Namespace Reference

Namespaces

- complex_literals

Classes

- class cmatrix
- class cmatrix_view
- class complex

  Wrapper class for gsl_complex structs.
- class complex_ref

  Stores a refernce to a gsl::complex object.
- class cvector

  A wrapper class for gsl_vector_complex.
- class cvector_view
- class crow_view

  A subclass of cvector_view for stride-1 cvectors.
- class ccolumn_view
- class matrix

  A wrapper class for gsl_matrix.
- class matrix_view
- class vector

  A wrapper class for gsl_vector.
- class vector_view
- class row_view

  A subclass of cvector_view for stride-1 cvectors.
- class column_view

  A subclass of vector_view for non-stride-1 vectors.

Enumerations

- enum class legendre_norm : int { none = GSL_SF_LEGENDRE_NONE , schmidt = GSL_SF_LEGENDRE↩ _SCHMIDT , spharm = GSL_SF_LEGENDRE_SPHARM , full = GSL_SF_LEGENDRE_FULL }

  Wrapper enum for the GSL gsl_sf_legendre_t.

Functions

- complex operator""_i (long double y)

  User defined literal overload for complex numbers.
- gsl::cvector fft (gsl::cvector &&x)

  Compute in-place fft of a gsl::(c)vector.
- gsl::cvector fft (const gsl::cvector &x)

  Compute fft of a gsl::(c)vector.
- gsl::cvector ifft (gsl::cvector &&x)

  Compute in-place inverse fft of a gsl::(c)vector.
- gsl::cvector ifft (const gsl::cvector &x)

  Compute inverse fft of a gsl::(c)vector.
- gsl::cmatrix fft (gsl::cmatrix &&x, int dim=1)

  Compute in-place 1D fft of each column/row of a gsl::(c)matrix.
- gsl::cmatrix fft (const gsl::cmatrix &x, int dim=1)

  Compute 1D fft of each column/row of a gsl::(c)matrix.
- gsl::cmatrix ifft (gsl::cmatrix &&x, int dim=1)

  Compute in-place 1D inverse fft of each column/row of a gsl::(c)matrix.
- gsl::cmatrix ifft (const gsl::cmatrix &x, int dim=1)

  Compute 1D inverse fft of each column/row of a gsl::(c)matrix.
- double spherical_harmonic (int n, int m, double x)

  Compute the normalized associated Legendre polynomial P_n^m(x)
- complex spherical_harmonic (int n, int m, double theta, double phi)

  Compute the normalized associated spherical harmonic Y_n^m(theta, phi)
- void fit_linear (gsl::vector &x, gsl::vector &y, double &c0, double &c1, double &sumsq)

  Compute the intercept `c0` and slope `c1` of best fit
- void leggauss (size_t n, gsl::vector &x, gsl::vector &w, double a=-1.0, double b=1.0)

  Store Gauss-Legendre quadrature nodes and weights.
- vector leggauss (size_t n, double a=-1.0, double b=1.0)

  Get a vector Gauss-Legendre quadrature nodes.
- gsl::vector linspace (double a, double b, size_t N=100)

  Get a gsl::vector of evenly spaced points on the interval [a, b] (inclusive)
- gsl::cvector linspace (gsl::complex a, gsl::complex b, size_t n)

  Complex version of linspace.
- gsl::vector arange (double a, double b, double step=1.0)

  Get a gsl::vector of `step` spaced points on the interval [`a, b`)
- gsl::vector circshift (const gsl::vector &x, int k)

  Perform a circular shift of the elements of a gsl::vector.
- gsl::cvector circshift (const gsl::cvector &x, int k)

  Perform a circular shift of the elements of a gsl::cvector.
- void meshgrid (const gsl::vector &x, const gsl::vector &y, gsl::matrix &X, gsl::matrix &Y)

  Store 2D grid coordinates based on 1D input gsl::vectors.
- void meshgrid (const gsl::cvector &x, const gsl::cvector &y, gsl::cmatrix &X, gsl::cmatrix &Y)

  Complex version of gsl::meshgrid.
- gsl::matrix eye (size_t n)

  Return the nxn identity matrix.
- template<typename Lambda >
  gsl::vector arrayfun (Lambda &&func, const gsl::vector &x)

  Apply lambda function to each element of a gsl::vector, akin to MATLAB arrayfun.

- template<typename Lambda >
  gsl::vector arrayfun (Lambda &&func, gsl::vector &&x)

  Move version of arrayfun for vectors.
- template<typename Lambda >
  gsl::cvector arrayfun (Lambda &&func, const gsl::cvector &x)

  Copy version of arrayfun for complex vectors.
- template<typename Lambda >
  gsl::cvector arrayfun (Lambda &&func, gsl::cvector &&x)

  Move version of arrayfun for complex vectors.
- template<typename Lambda >
  gsl::matrix arrayfun (Lambda &&func, const gsl::matrix &x)

  Apply lambda function to each element of a gsl::matrix, akin to MATLAB arrayfun.
- template<typename Lambda >
  gsl::matrix arrayfun (Lambda &&func, gsl::matrix &&x)

  Move version of arrayfun.
- template<typename Lambda >
  gsl::cmatrix arrayfun (Lambda &&func, const gsl::cmatrix &x)

  Copy version of arrayfun for complex matrices.
- template<typename Lambda >
  gsl::cmatrix arrayfun (Lambda &&func, gsl::cmatrix &&x)

  Move version of arrayfun for complex matrices.
- gsl::vector diag (const gsl::matrix &A)
- gsl::vector pow (const gsl::vector &x, double p)
- gsl::cvector pow (const gsl::cvector &x, complex p)
- gsl::matrix pow (const gsl::matrix &x, double p)
- gsl::cmatrix pow (const gsl::cmatrix &x, complex p)
- cmatrix operator∗ (double a, const cmatrix &M)
- cmatrix operator∗ (const cmatrix &M, double a)
- cmatrix operator∗ (double a, cmatrix &&M)
- cmatrix operator∗ (cmatrix &&M, double a)
- cmatrix operator∗ (const cmatrix &M, const complex &a)
- cmatrix operator∗ (const complex &a, const cmatrix &M)
- cmatrix operator∗ (complex z, const cmatrix &M)
- cmatrix operator∗ (const cmatrix &M, complex z)
- cmatrix operator∗ (cmatrix &&M, complex z)
- cmatrix operator∗ (complex z, cmatrix &&M)
- cmatrix operator∗ (complex a, const matrix &M)
- cmatrix operator∗ (const matrix &M, complex a)
- cmatrix operator/ (double a, const cmatrix &M)
- cmatrix operator/ (const cmatrix &M, double a)
- cmatrix operator/ (double a, cmatrix &&M)
- cmatrix operator/ (cmatrix &&M, double a)
- cmatrix operator/ (const cmatrix &M, const complex &a)
- cmatrix operator/ (const complex &a, const cmatrix &M)
- cmatrix operator/ (complex z, const cmatrix &M)
- cmatrix operator/ (const cmatrix &M, complex z)
- cmatrix operator/ (cmatrix &&M, complex z)
- cmatrix operator/ (complex z, cmatrix &&M)
- cmatrix operator/ (complex z, const matrix &M)
- cmatrix operator/ (const matrix &M, complex z)
- cmatrix operator+ (const cmatrix &M1, const cmatrix &M2)
- cmatrix operator+ (cmatrix &&M1, const cmatrix &M2)
- cmatrix operator+ (const cmatrix &M1, cmatrix &&M2)
- cmatrix operator+ (cmatrix &&M1, cmatrix &&M2)

- cmatrix operator+ (const cmatrix &M1, const matrix &M2)
- cmatrix operator+ (cmatrix &&M1, const matrix &M2)
- cmatrix operator+ (const matrix &M1, const cmatrix &M2)
- cmatrix operator+ (const matrix &M1, cmatrix &&M2)
- cmatrix operator- (const cmatrix &M1, const cmatrix &M2)
- cmatrix operator- (cmatrix &&M1, const cmatrix &M2)
- cmatrix operator- (const cmatrix &M1, cmatrix &&M2)
- cmatrix operator- (cmatrix &&M1, cmatrix &&M2)
- cmatrix operator- (const cmatrix &M1, const matrix &M2)
- cmatrix operator- (cmatrix &&M1, const matrix &M2)
- cmatrix operator- (const matrix &M1, const cmatrix &M2)
- cmatrix operator- (const matrix &M1, cmatrix &&M2)
- cmatrix operator∗ (const cmatrix &A, const cmatrix &B)
- bool operator== (const cmatrix &M1, const cmatrix &M2)
- bool operator!= (const cmatrix &M1, const cmatrix &M2)
- bool operator== (const cmatrix &M1, const matrix &M2)
- bool operator!= (const cmatrix &M1, const matrix &M2)
- bool operator== (const matrix &M1, const cmatrix &M2)
- bool operator!= (const matrix &M1, const cmatrix &M2)
- cvector operator∗ (complex z, const cvector &v)
- cvector operator∗ (complex z, cvector &&v)
- cvector operator∗ (const cvector &v, complex z)
- cvector operator∗ (cvector &&v, complex z)
- cvector operator∗ (complex a, const vector &v)
- cvector operator∗ (const vector &v, complex a)
- cvector operator∗ (const cvector &v, double x)
- cvector operator∗ (double x, const cvector &v)
- cvector operator∗ (cvector &&v, double x)
- cvector operator∗ (double x, cvector &&v)
- cvector operator/ (const cvector &v, complex z)
- cvector operator/ (cvector &&v, complex z)
- cvector operator/ (complex z, const cvector &v)
- cvector operator/ (complex z, cvector &&v)
- cvector operator/ (const vector &v, complex z)
- cvector operator/ (complex z, const vector &v)
- cvector operator/ (const cvector &v, double x)
- cvector operator/ (cvector &&v, double x)
- cvector operator/ (double x, const cvector &v)
- cvector operator/ (double x, cvector &&v)
- cvector operator+ (const cvector &v1, const cvector &v2)
- cvector operator+ (cvector &&v1, const cvector &v2)
- cvector operator+ (const cvector &v1, cvector &&v2)
- cvector operator+ (cvector &&v1, cvector &&v2)
- cvector operator- (const cvector &v1, const cvector &v2)
- cvector operator- (cvector &&v1, const cvector &v2)
- cvector operator- (const cvector &v1, cvector &&v2)
- cvector operator- (cvector &&v1, cvector &&v2)
- cvector operator- (const vector &v1, const cvector &v2)
- cvector operator- (const vector &v1, cvector &&v2)
- cvector operator- (const cvector &v1, const vector &v2)
- cvector operator- (cvector &&v1, const vector &v2)
- bool operator== (const cvector &v1, const cvector &v2)
- bool operator!= (const cvector &v1, const cvector &v2)
- bool operator== (const vector &v1, const cvector &v2)
- bool operator!= (const vector &v1, const cvector &v2)

- bool operator== (const cvector &v1, const vector &v2)
- bool operator!= (const cvector &v1, const vector &v2)
- cvector operator* (const cmatrix &M, const cvector &v)
- matrix operator* (double a, const matrix &M)
- matrix operator* (double a, matrix &&M)
- matrix operator* (const matrix &M, double a)
- matrix operator* (matrix &&M, double a)
- matrix operator/ (double a, const matrix &M)
- matrix operator/ (double a, matrix &&M)
- matrix operator/ (const matrix &M, double a)
- matrix operator/ (matrix &&M, double a)
- matrix operator+ (const matrix &M1, const matrix &M2)
- matrix operator+ (const matrix &M1, matrix &&M2)
- matrix operator+ (matrix &&M1, const matrix &M2)
- matrix operator+ (matrix &&M1, matrix &&M2)
- matrix operator- (const matrix &M1, const matrix &M2)
- matrix operator- (const matrix &M1, matrix &&M2)
- matrix operator- (matrix &&M1, const matrix &M2)
- matrix operator- (matrix &&M1, matrix &&M2)
- matrix operator* (const matrix &A, const matrix &B)
- bool operator== (const matrix &M1, const matrix &M2)
- bool operator!= (const matrix &M1, const matrix &M2)
- vector operator* (double a, const vector &v)
- vector operator* (double a, vector &&v)
- vector operator* (const vector &v, double a)
- vector operator* (vector &&v, double a)
- vector operator/ (double a, const vector &v)
- vector operator/ (double a, vector &&v)
- vector operator/ (const vector &v, double a)
- vector operator/ (vector &&v, double a)
- vector operator+ (const vector &v1, const vector &v2)
- vector operator+ (vector &&v1, const vector &v2)
- vector operator+ (const vector &v1, vector &&v2)
- vector operator+ (vector &&v1, vector &&v2)
- cvector operator+ (const vector &v1, const cvector &v2)
- cvector operator+ (const vector &v1, cvector &&v2)
- cvector operator+ (const cvector &v1, const vector &v2)
- cvector operator+ (cvector &&v1, const vector &v2)
- vector operator- (const vector &v1, const vector &v2)
- vector operator- (vector &&v1, const vector &v2)
- vector operator- (const vector &v1, vector &&v2)
- vector operator- (vector &&v1, vector &&v2)
- bool operator== (const vector &v1, const vector &v2)
- bool operator!= (const vector &v1, const vector &v2)
- vector operator* (const matrix &M, const vector &v)

6.1.1 Enumeration Type Documentation

6.1.1.1 legendre_norm

```
enum gsl::legendre_norm : int [strong]
```

Wrapper enum for the GSL gsl_sf_legendre_t.

Enumerator

none	
schmidt	
spharm	
full	

6.1.2 Function Documentation

6.1.2.1 arange()

```
gsl::vector gsl::arange (
            double start,
            double stop,
            double step = 1.0 )
```

Get a gsl::vector of `step` spaced points on the interval [`a`, `b`)

Parameters

a	Lower bound of the interval
b	Upper bound of the interval
step	Spacing between points

Note

Will return an empty vector if `b` cannot be reached by steping from `a` by `step`.

Returns

gsl::vector of points

6.1.2.2 arrayfun() [1/8]

```
template<typename Lambda >
gsl::cmatrix gsl::arrayfun (
            Lambda && func,
            const gsl::cmatrix & x )
```

Copy version of arrayfun for complex matrices.

6.1.2.3 arrayfun() [2/8]

```
template<typename Lambda >
gsl::cvector gsl::arrayfun (
             Lambda && func,
             const gsl::cvector & x )
```

Copy version of arrayfun for complex vectors.

6.1.2.4 arrayfun() [3/8]

```
template<typename Lambda >
gsl::matrix gsl::arrayfun (
             Lambda && func,
             const gsl::matrix & x )
```

Apply lambda function to each element of a gsl::matrix, akin to MATLAB arrayfun.

Parameters

func	Lambda function to apply to each element
x	Input matrix

Note

This function is not optimized for speed, but rather for convenience, as there is overhead in using templated Lambda functions. If speed is necessary, add the function to gsl_utils specifically.

Returns

Matrix of same size as `x` with `func` applied to each element

6.1.2.5 arrayfun() [4/8]

```
template<typename Lambda >
gsl::vector gsl::arrayfun (
             Lambda && func,
             const gsl::vector & x )
```

Apply lambda function to each element of a gsl::vector, akin to MATLAB arrayfun.

Parameters

func	Lambda function to apply to each element
x	Input matrix

Note

This function is not optimized for speed, but rather for convenience, as there is overhead in using templated Lambda functions. If speed is necessary, add the function to gsl_utils specifically.

Returns

Matrix of same size as `x` with `func` applied to each element

6.1.2.6 arrayfun() [5/8]

```
template<typename Lambda >
gsl::cmatrix gsl::arrayfun (
            Lambda && func,
            gsl::cmatrix && x )
```

Move version of arrayfun for complex matrices.

6.1.2.7 arrayfun() [6/8]

```
template<typename Lambda >
gsl::cvector gsl::arrayfun (
            Lambda && func,
            gsl::cvector && x )
```

Move version of arrayfun for complex vectors.

6.1.2.8 arrayfun() [7/8]

```
template<typename Lambda >
gsl::matrix gsl::arrayfun (
            Lambda && func,
            gsl::matrix && x )
```

Move version of arrayfun.

6.1.2.9 arrayfun() [8/8]

```
template<typename Lambda >
gsl::vector gsl::arrayfun (
            Lambda && func,
            gsl::vector && x )
```

Move version of arrayfun for vectors.

6.1.2.10 circshift() [1/2]

```
gsl::cvector gsl::circshift (
            const gsl::cvector & x,
            int k )
```

Perform a circular shift of the elements of a gsl::cvector.

Parameters

x	Input vector
k	Number of positions to shift

Returns

Shifted vector

6.1.2.11 circshift() [2/2]

```
gsl::vector gsl::circshift (
            const gsl::vector & x,
            int k )
```

Perform a circular shift of the elements of a gsl::vector.

Parameters

x	Input vector
k	Number of positions to shift

Returns

Shifted vector

6.1.2.12 diag()

```
gsl::vector gsl::diag (
            const gsl::matrix & A )
```

6.1.2.13 eye()

```
gsl::matrix gsl::eye (
            size_t n )
```

Return the nxn identity matrix.

6.1.2.14 fft() [1/4]

```
gsl::cmatrix gsl::fft (
            const gsl::cmatrix & x,
            int dim = 1 )
```

Compute 1D fft of each column/row of a gsl::(c)matrix.

Compute 1D fft of each column/row of a gsl::(c)matrix Computes the 1D complex Fast Fourier Transform of each row (dim=1) or column (dim=2) of the data in x.

Parameters

x	The cmatrix to be transformed.
dim	The dimension to transform. 1 for columns, 2 for rows.

Returns

The transformed matrix

6.1.2.15 fft() [2/4]

```
gsl::cvector gsl::fft (
            const gsl::cvector & x )
```

Compute fft of a gsl::(c)vector.

Compute fft of a gsl::(c)vector Computes the complex Fast Fourier Transform of the data in x.

Parameters

x	The cvector to be transformed.

Returns

The transformed vector

6.1.2.16 fft() [3/4]

```
gsl::cmatrix gsl::fft (
            gsl::cmatrix && x,
            int dim = 1 )
```

Compute in-place 1D fft of each column/row of a gsl::(c)matrix.

Compute in-place 1D fft of each column/row of a gsl::(c)matrix Computes the 1D complex Fast Fourier Transform of each row (dim=1) or column (dim=2) of the data in x.

Parameters

x	The cmatrix to be transformed.
dim	The dimension to transform. 1 for columns, 2 for rows.

Returns

The transformed matrix, reclaiming memory of the original matrix

6.1.2.17 fft() [4/4]

```
gsl::cvector gsl::fft (
            gsl::cvector && x )
```

Compute in-place fft of a gsl::(c)vector.

Compute in-place fft of a gsl::(c)vector Computes the in-place complex Fast Fourier Transform of the data in x.

Parameters

x	The cvector to be transformed.

Note

More efficient implementations of this function (and the others in this file) would use gsl_fft_real_* functions for real arguments, or cache the wavetable and workspace for repeated calls in different parts of execution.

Returns

The transformed vector, reclaiming the memory of the original vector.

6.1.2.18 fit_linear()

```
void gsl::fit_linear (
            gsl::vector & x,
            gsl::vector & y,
            double & c0,
            double & c1,
            double & sumsq )
```

Compute the intercept `c0` and slope `c1` of best fit

Compute in-place fft of a gsl::(c)vector Computes the in-place complex Fast Fourier Transform of the data in x.

Parameters

x	The cvector to be transformed.

Note

More efficient implementations of this function (and the others in this file) would use gsl_fft_real_* functions for real arguments, or cache the wavetable and workspace for repeated calls in different parts of execution.

Returns

The transformed vector, reclaiming the memory of the original vector.

6.1.2.19 ifft() [1/4]

```
gsl::cmatrix gsl::ifft (
            const gsl::cmatrix & x,
            int dim = 1 )
```

Compute 1D inverse fft of each column/row of a gsl::(c)matrix.

Compute 1D inverse fft of each column/row of a gsl::(c)matrix Computes the 1D complex inverse Fast Fourier Transform of each row (dim=1) or column (dim=2) of the data in x.

Parameters

x	The cmatrix to be transformed.
dim	The dimension to transform. 1 for columns, 2 for rows.

Returns

The transformed matrix

6.1.2.20 ifft() [2/4]

```
gsl::cvector gsl::ifft (
            const gsl::cvector & x )
```

Compute inverse fft of a gsl::(c)vector.

Compute inverse fft of a gsl::(c)vector Computes the complex inverse Fast Fourier Transform of the data in x.

Parameters

x	The cvector to be transformed.

Returns

The transformed vector

6.1.2.21 ifft() [3/4]

```
gsl::cmatrix gsl::ifft (
            gsl::cmatrix && x,
            int dim = 1 )
```

Compute in-place 1D inverse fft of each column/row of a gsl::(c)matrix.

Compute in-place 1D inverse fft of each column/row of a gsl::(c)matrix Computes the 1D complex inverse Fast Fourier Transform of each row (dim=1) or column (dim=2) of the data in x.

Parameters

x	The cmatrix to be transformed.
dim	The dimension to transform. 1 for columns, 2 for rows.

Returns

The transformed matrix, reclaiming memory of the original matrix

6.1.2.22 ifft() [4/4]

```
gsl::cvector gsl::ifft (
            gsl::cvector && x )
```

Compute in-place inverse fft of a gsl::(c)vector.

Compute in-place inverse fft of a gsl::(c)vector Computes the in-place complex Inverse Fast Fourier Transform of the data in x.

Parameters

x	The cvector to be transformed.

Returns

The transformed vector, reclaiming the memory of the original vector.

6.1.2.23 leggauss() [1/2]

```
gsl::vector gsl::leggauss (
            size_t n,
            double a = -1.0,
            double b = 1.0 )
```

Get a vector Gauss-Legendre quadrature nodes.

Computes n Gauss-Legendre quadrature nodes and weights for the interval [a, b] (default [-1, 1])

Parameters

n	Number of nodes
a	Lower bound of the interval (Default -1)
b	Upper bound of the interval (Default +1)

Returns

gsl::vector of nodes

6.1.2.24 leggauss() [2/2]

```
void gsl::leggauss (
            size_t n,
            gsl::vector & x,
            gsl::vector & w,
            double a = -1.0,
            double b = 1.0 )
```

Store Gauss-Legendre quadrature nodes and weights.

Computes n Gauss-Legendre quadrature nodes and weights for the interval [a, b] (default [-1, 1])

Parameters

n	Number of nodes
x	Reference to a gsl::vector of nodes
w	Reference to a gsl::vector of weights
a	Lower bound of the interval (Default -1)
b	Upper bound of the interval (Default +1)

Note

This could be made more efficient by caching the gsl_integration_glfixed_table, rather than allocating and freeing it each time.

6.1.2.25 linspace() [1/2]

```
gsl::vector gsl::linspace (
            double a,
            double b,
            size_t n = 100 )
```

Get a gsl::vector of evenly spaced points on the interval [a, b] (inclusive)

Computes n evenly spaced points on the interval [a, b] (inclusive)

Parameters

a	Lower bound of the interval
b	Upper bound of the interval
n	Number of points

Returns

gsl::vector of points

6.1.2.26 linspace() [2/2]

```
gsl::cvector gsl::linspace (
            gsl::complex a,
            gsl::complex b,
            size_t n )
```

Complex version of linspace.

6.1.2.27 meshgrid() [1/2]

```
void gsl::meshgrid (
            const gsl::cvector & x,
            const gsl::cvector & y,
            gsl::cmatrix & X,
            gsl::cmatrix & Y )
```

Complex version of gsl::meshgrid.

Store 2D grid coordinates based on 1D input gsl::cvectors.

Parameters

x	1D complex vector of x-coordinates
y	1D complex vector of y-coordinates
X	2D complex matrix of x-coordinates
Y	2D complex matrix of y-coordinates

6.1.2.28 meshgrid() [2/2]

```
void gsl::meshgrid (
            const gsl::vector & x,
            const gsl::vector & y,
            gsl::matrix & X,
            gsl::matrix & Y )
```

Store 2D grid coordinates based on 1D input gsl::vectors.

Parameters

x	1D vector of x-coordinates
y	1D vector of y-coordinates
X	2D matrix of x-coordinates
Y	2D matrix of y-coordinates

6.1.2.29 operator"!=() [1/8]

```
bool gsl::operator!= (
            const cmatrix & M1,
            const cmatrix & M2 )
```

6.1.2.30 operator"!=() [2/8]

```
bool gsl::operator!= (
            const cmatrix & M1,
            const matrix & M2 )
```

6.1.2.31 operator"!=() [3/8]

```
bool gsl::operator!= (
            const cvector & v1,
            const cvector & v2 )
```

6.1.2.32 operator"!=() [4/8]

```
bool gsl::operator!= (
            const cvector & v1,
            const vector & v2 )
```

6.1.2.33 operator"!=() [5/8]

```
bool gsl::operator!= (
            const matrix & M1,
            const cmatrix & M2 )
```

6.1.2.34 operator"!=() [6/8]

```
bool gsl::operator!= (
            const matrix & M1,
            const matrix & M2 )
```

6.1.2.35 operator"!=() [7/8]

```
bool gsl::operator!= (
            const vector & v1,
            const cvector & v2 )
```

6.1.2.36 operator"!=() [8/8]

```
bool gsl::operator!= (
            const vector & v1,
            const vector & v2 )
```

6.1.2.37 operator*() [1/34]

```
cmatrix gsl::operator* (
            cmatrix && M,
            complex z )
```

6.1.2.38 operator*() [2/34]

```
cmatrix gsl::operator* (
            cmatrix && M,
            double a )
```

6.1.2.39 operator*() [3/34]

```
cmatrix gsl::operator* (
            complex a,
            const matrix & M )
```

6.1.2.40 operator*() [4/34]

```
cvector gsl::operator* (
            complex a,
            const vector & v )
```

6.1.2.41 operator*() [5/34]

```
cmatrix gsl::operator* (
            complex z,
            cmatrix && M )
```

6.1.2.42 operator*() [6/34]

```
cmatrix gsl::operator* (
            complex z,
            const cmatrix & M )
```

6.1.2.43 operator*() [7/34]

```
cvector gsl::operator* (
            complex z,
            const cvector & v )
```

6.1.2.44 operator*() [8/34]

```
cvector gsl::operator* (
            complex z,
            cvector && v )
```

6.1.2.45 operator*() [9/34]

```
cmatrix gsl::operator* (
            const cmatrix & A,
            const cmatrix & B )
```

6.1.2.46 operator*() [10/34]

```
cmatrix gsl::operator* (
            const cmatrix & M,
            complex z )
```

6.1.2.47 operator*() [11/34]

```
cmatrix gsl::operator* (
            const cmatrix & M,
            const complex & a )
```

6.1.2.48 operator*() [12/34]

```
cvector gsl::operator* (
            const cmatrix & M,
            const cvector & v )
```

6.1.2.49 operator*() [13/34]

```
cmatrix gsl::operator* (
            const cmatrix & M,
            double a )
```

6.1.2.50 operator∗() [14/34]

```
cmatrix gsl::operator* (
            const complex & a,
            const cmatrix & M )
```

6.1.2.51 operator∗() [15/34]

```
cvector gsl::operator* (
            const cvector & v,
            complex z )
```

6.1.2.52 operator∗() [16/34]

```
cvector gsl::operator* (
            const cvector & v,
            double x )
```

6.1.2.53 operator∗() [17/34]

```
matrix gsl::operator* (
            const matrix & A,
            const matrix & B )
```

6.1.2.54 operator∗() [18/34]

```
cmatrix gsl::operator* (
            const matrix & M,
            complex a )
```

6.1.2.55 operator∗() [19/34]

```
vector gsl::operator* (
            const matrix & M,
            const vector & v )
```

6.1.2.56 operator∗() [20/34]

```
matrix gsl::operator* (
            const matrix & M,
            double a )
```

6.1.2.57 operator∗() [21/34]

```
cvector gsl::operator* (
            const vector & v,
            complex a )
```

6.1.2.58 operator∗() [22/34]

```
vector gsl::operator* (
            const vector & v,
            double a )
```

6.1.2.59 operator∗() [23/34]

```
cvector gsl::operator* (
            cvector && v,
            complex z )
```

6.1.2.60 operator∗() [24/34]

```
cvector gsl::operator* (
            cvector && v,
            double x )
```

6.1.2.61 operator∗() [25/34]

```
cmatrix gsl::operator* (
            double a,
            cmatrix && M )
```

6.1.2.62 operator*() [26/34]

```
cmatrix gsl::operator* (
            double a,
            const cmatrix & M )
```

6.1.2.63 operator*() [27/34]

```
matrix gsl::operator* (
            double a,
            const matrix & M )
```

6.1.2.64 operator*() [28/34]

```
vector gsl::operator* (
            double a,
            const vector & v )
```

6.1.2.65 operator*() [29/34]

```
matrix gsl::operator* (
            double a,
            matrix && M )
```

6.1.2.66 operator*() [30/34]

```
vector gsl::operator* (
            double a,
            vector && v )
```

6.1.2.67 operator*() [31/34]

```
cvector gsl::operator* (
            double x,
            const cvector & v )
```

6.1.2.68 operator∗() [32/34]

```
cvector gsl::operator* (
            double x,
            cvector && v )
```

6.1.2.69 operator∗() [33/34]

```
matrix gsl::operator* (
            matrix && M,
            double a )
```

6.1.2.70 operator∗() [34/34]

```
vector gsl::operator* (
            vector && v,
            double a )
```

6.1.2.71 operator+() [1/24]

```
cmatrix gsl::operator+ (
            cmatrix && M1,
            cmatrix && M2 )
```

6.1.2.72 operator+() [2/24]

```
cmatrix gsl::operator+ (
            cmatrix && M1,
            const cmatrix & M2 )
```

6.1.2.73 operator+() [3/24]

```
cmatrix gsl::operator+ (
            cmatrix && M1,
            const matrix & M2 )
```

6.1.2.74 operator+() [4/24]

```
cmatrix gsl::operator+ (
            const cmatrix & M1,
            cmatrix && M2 )
```

6.1.2.75 operator+() [5/24]

```
cmatrix gsl::operator+ (
            const cmatrix & M1,
            const cmatrix & M2 )
```

6.1.2.76 operator+() [6/24]

```
cmatrix gsl::operator+ (
            const cmatrix & M1,
            const matrix & M2 )
```

6.1.2.77 operator+() [7/24]

```
cvector gsl::operator+ (
            const cvector & v1,
            const cvector & v2 )
```

6.1.2.78 operator+() [8/24]

```
cvector gsl::operator+ (
            const cvector & v1,
            const vector & v2 )
```

6.1.2.79 operator+() [9/24]

```
cvector gsl::operator+ (
            const cvector & v1,
            cvector && v2 )
```

6.1.2.80 operator+() [10/24]

```
cmatrix gsl::operator+ (
            const matrix & M1,
            cmatrix && M2 )
```

6.1.2.81 operator+() [11/24]

```
cmatrix gsl::operator+ (
            const matrix & M1,
            const cmatrix & M2 )
```

6.1.2.82 operator+() [12/24]

```
matrix gsl::operator+ (
            const matrix & M1,
            const matrix & M2 )
```

6.1.2.83 operator+() [13/24]

```
matrix gsl::operator+ (
            const matrix & M1,
            matrix && M2 )
```

6.1.2.84 operator+() [14/24]

```
cvector gsl::operator+ (
            const vector & v1,
            const cvector & v2 )
```

6.1.2.85 operator+() [15/24]

```
vector gsl::operator+ (
            const vector & v1,
            const vector & v2 )
```

6.1.2.86 operator+() [16/24]

```
cvector gsl::operator+ (
            const vector & v1,
            cvector && v2 )
```

6.1.2.87 operator+() [17/24]

```
vector gsl::operator+ (
            const vector & v1,
            vector && v2 )
```

6.1.2.88 operator+() [18/24]

```
cvector gsl::operator+ (
            cvector && v1,
            const cvector & v2 )
```

6.1.2.89 operator+() [19/24]

```
cvector gsl::operator+ (
            cvector && v1,
            const vector & v2 )
```

6.1.2.90 operator+() [20/24]

```
cvector gsl::operator+ (
            cvector && v1,
            cvector && v2 )
```

6.1.2.91 operator+() [21/24]

```
matrix gsl::operator+ (
            matrix && M1,
            const matrix & M2 )
```

6.1.2.92 operator+() [22/24]

```
matrix gsl::operator+ (
            matrix && M1,
            matrix && M2 )
```

6.1.2.93 operator+() [23/24]

```
vector gsl::operator+ (
            vector && v1,
            const vector & v2 )
```

6.1.2.94 operator+() [24/24]

```
vector gsl::operator+ (
            vector && v1,
            vector && v2 )
```

6.1.2.95 operator-() [1/24]

```
cmatrix gsl::operator- (
            cmatrix && M1,
            cmatrix && M2 )
```

6.1.2.96 operator-() [2/24]

```
cmatrix gsl::operator- (
            cmatrix && M1,
            const cmatrix & M2 )
```

6.1.2.97 operator-() [3/24]

```
cmatrix gsl::operator- (
            cmatrix && M1,
            const matrix & M2 )
```

6.1.2.98 operator-() [4/24]

```
cmatrix gsl::operator- (
            const cmatrix & M1,
            cmatrix && M2 )
```

6.1.2.99 operator-() [5/24]

```
cmatrix gsl::operator- (
            const cmatrix & M1,
            const cmatrix & M2 )
```

6.1.2.100 operator-() [6/24]

```
cmatrix gsl::operator- (
            const cmatrix & M1,
            const matrix & M2 )
```

6.1.2.101 operator-() [7/24]

```
cvector gsl::operator- (
            const cvector & v1,
            const cvector & v2 )
```

6.1.2.102 operator-() [8/24]

```
cvector gsl::operator- (
            const cvector & v1,
            const vector & v2 )
```

6.1.2.103 operator-() [9/24]

```
cvector gsl::operator- (
            const cvector & v1,
            cvector && v2 )
```

6.1.2.104 operator-() [10/24]

```
cmatrix gsl::operator- (
            const matrix & M1,
            cmatrix && M2 )
```

6.1.2.105 operator-() [11/24]

```
cmatrix gsl::operator- (
            const matrix & M1,
            const cmatrix & M2 )
```

6.1.2.106 operator-() [12/24]

```
matrix gsl::operator- (
            const matrix & M1,
            const matrix & M2 )
```

6.1.2.107 operator-() [13/24]

```
matrix gsl::operator- (
            const matrix & M1,
            matrix && M2 )
```

6.1.2.108 operator-() [14/24]

```
cvector gsl::operator- (
            const vector & v1,
            const cvector & v2 )
```

6.1.2.109 operator-() [15/24]

```
vector gsl::operator- (
            const vector & v1,
            const vector & v2 )
```

6.1.2.110 operator-() [16/24]

```
cvector gsl::operator- (
            const vector & v1,
            cvector && v2 )
```

6.1.2.111 operator-() [17/24]

```
vector gsl::operator- (
            const vector & v1,
            vector && v2 )
```

6.1.2.112 operator-() [18/24]

```
cvector gsl::operator- (
            cvector && v1,
            const cvector & v2 )
```

6.1.2.113 operator-() [19/24]

```
cvector gsl::operator- (
            cvector && v1,
            const vector & v2 )
```

6.1.2.114 operator-() [20/24]

```
cvector gsl::operator- (
            cvector && v1,
            cvector && v2 )
```

6.1.2.115 operator-() [21/24]

```
matrix gsl::operator- (
            matrix && M1,
            const matrix & M2 )
```

6.1.2.116 operator-() [22/24]

```
matrix gsl::operator- (
            matrix && M1,
            matrix && M2 )
```

6.1.2.117 operator-() [23/24]

```
vector gsl::operator- (
            vector && v1,
            const vector & v2 )
```

6.1.2.118 operator-() [24/24]

```
vector gsl::operator- (
            vector && v1,
            vector && v2 )
```

6.1.2.119 operator/() [1/30]

```
cmatrix gsl::operator/ (
            cmatrix && M,
            complex z )
```

6.1.2.120 operator/() [2/30]

```
cmatrix gsl::operator/ (
            cmatrix && M,
            double a )
```

6.1.2.121 operator/() [3/30]

```
cmatrix gsl::operator/ (
            complex z,
            cmatrix && M )
```

6.1.2.122 operator/() [4/30]

```
cmatrix gsl::operator/ (
            complex z,
            const cmatrix & M )
```

6.1.2.123 operator/() [5/30]

```
cvector gsl::operator/ (
            complex z,
            const cvector & v )
```

6.1.2.124 operator/() [6/30]

```
cmatrix gsl::operator/ (
            complex z,
            const matrix & M )
```

6.1.2.125 operator/() [7/30]

```
cvector gsl::operator/ (
            complex z,
            const vector & v )
```

6.1.2.126 operator/() [8/30]

```
cvector gsl::operator/ (
            complex z,
            cvector && v )
```

6.1.2.127 operator/() [9/30]

```
cmatrix gsl::operator/ (
            const cmatrix & M,
            complex z )
```

6.1.2.128 operator/() [10/30]

```
cmatrix gsl::operator/ (
            const cmatrix & M,
            const complex & a )
```

6.1.2.129 operator/() [11/30]

```
cmatrix gsl::operator/ (
            const cmatrix & M,
            double a )
```

6.1.2.130 operator/() [12/30]

```
cmatrix gsl::operator/ (
            const complex & a,
            const cmatrix & M )
```

6.1.2.131 operator/() [13/30]

```
cvector gsl::operator/ (
            const cvector & v,
            complex z )
```

6.1.2.132 operator/() [14/30]

```
cvector gsl::operator/ (
            const cvector & v,
            double x )
```

6.1.2.133 operator/() [15/30]

```
cmatrix gsl::operator/ (
            const matrix & M,
            complex z )
```

6.1.2.134 operator/() [16/30]

```
matrix gsl::operator/ (
            const matrix & M,
            double a )
```

6.1.2.135 operator/() [17/30]

```
cvector gsl::operator/ (
            const vector & v,
            complex z )
```

6.1.2.136 operator/() [18/30]

```
vector gsl::operator/ (
            const vector & v,
            double a )
```

6.1.2.137 operator/() [19/30]

```
cvector gsl::operator/ (
            cvector && v,
            complex z )
```

6.1.2.138 operator/() [20/30]

```
cvector gsl::operator/ (
            cvector && v,
            double x )
```

6.1.2.139 operator/() [21/30]

```
cmatrix gsl::operator/ (
            double a,
            cmatrix && M )
```

6.1.2.140 operator/() [22/30]

```
cmatrix gsl::operator/ (
            double a,
            const cmatrix & M )
```

6.1.2.141 operator/() [23/30]

```
matrix gsl::operator/ (
            double a,
            const matrix & M )
```

6.1.2.142 operator/() [24/30]

```
vector gsl::operator/ (
            double a,
            const vector & v )
```

6.1.2.143 operator/() [25/30]

```
matrix gsl::operator/ (
            double a,
            matrix && M )
```

6.1.2.144 operator/() [26/30]

```
vector gsl::operator/ (
            double a,
            vector && v )
```

6.1.2.145 operator/() [27/30]

```
cvector gsl::operator/ (
            double x,
            const cvector & v )
```

6.1.2.146 operator/() [28/30]

```
cvector gsl::operator/ (
            double x,
            cvector && v )
```

6.1.2.147 operator/() [29/30]

```
matrix gsl::operator/ (
            matrix && M,
            double a )
```

6.1.2.148 operator/() [30/30]

```
vector gsl::operator/ (
            vector && v,
            double a )
```

6.1.2.149 operator==() [1/8]

```
bool gsl::operator== (
            const cmatrix & M1,
            const cmatrix & M2 )
```

6.1.2.150 operator==() [2/8]

```
bool gsl::operator== (
            const cmatrix & M1,
            const matrix & M2 )
```

6.1.2.151 operator==() [3/8]

```
bool gsl::operator== (
            const cvector & v1,
            const cvector & v2 )
```

6.1.2.152 operator==() [4/8]

```
bool gsl::operator== (
            const cvector & v1,
            const vector & v2 )
```

6.1.2.153 operator==() [5/8]

```
bool gsl::operator== (
            const matrix & M1,
            const cmatrix & M2 )
```

6.1.2.154 operator==() [6/8]

```
bool gsl::operator== (
            const matrix & M1,
            const matrix & M2 )
```

6.1.2.155 operator==() [7/8]

```
bool gsl::operator== (
            const vector & v1,
            const cvector & v2 )
```

6.1.2.156 operator==() [8/8]

```
bool gsl::operator== (
            const vector & v1,
            const vector & v2 )
```

6.1.2.157 pow() [1/4]

```
gsl::cmatrix gsl::pow (
            const gsl::cmatrix & x,
            gsl::complex p )
```

6.1.2.158 pow() [2/4]

```
gsl::cvector gsl::pow (
            const gsl::cvector & x,
            gsl::complex p )
```

6.1.2.159 pow() [3/4]

```
gsl::matrix gsl::pow (
            const gsl::matrix & x,
            double p )
```

6.1.2.160 pow() [4/4]

```
gsl::vector gsl::pow (
            const gsl::vector & x,
            double p )
```

6.1.2.161 spherical_harmonic() [1/2]

```
gsl::complex gsl::spherical_harmonic (
            int n,
            int m,
            double theta,
            double phi )
```

Compute the normalized associated spherical harmonic Y_n^m(theta, phi)

Parameters

n	Degree of the polynomial
m	Order of the polynomial
theta	Azimuthal angle
phi	Polar angle

Note

Assumes n >= 0, |m| <= n, and 0 <= theta <= pi, 0 <= phi <= 2pi

Returns

The value of the spherical harmonic at x

6.1.2.162 spherical_harmonic() [2/2]

```
double gsl::spherical_harmonic (
            int n,
            int m,
            double x )
```

Compute the normalized associated Legendre polynomial P_n^m(x)

Parameters

n	Degree of the polynomial
m	Order of the polynomial
x	Point at which to evaluate the polynomial

Note

Assumes n >= 0, |m| <= n, and -1 <= x <= 1

Returns

The value of the polynomial at x

6.2 gsl::complex_literals Namespace Reference

Functions

- complex operator""_i (long double y)

  User defined literal overload for complex numbers.

6.2.1 Function Documentation

6.2.1.1 operator""_i()

```
complex gsl::complex_literals::operator""_i (
            long double y )  [inline]
```

User defined literal overload for complex numbers.

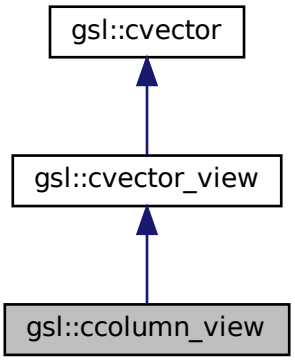
Chapter 7

Class Documentation

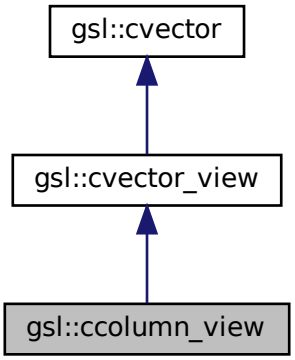
7.1 gsl::ccolumn_view Class Reference

`#include <cvector.h>`

Inheritance diagram for gsl::ccolumn_view:



Collaboration diagram for gsl::ccolumn_view:



Public Member Functions

- ccolumn_view (gsl_vector_complex ∗gvec_other)

  Construct column_view from existing vector view.
- ccolumn_view & operator= (const cvector &v)

  Assign data from complex vector to view.

Additional Inherited Members

7.1.1 Constructor & Destructor Documentation

7.1.1.1 ccolumn_view()

```
gsl::ccolumn_view::ccolumn_view (
            gsl_vector_complex * gvec_other )  [inline]
```

Construct column_view from existing vector view.

7.1.2 Member Function Documentation

7.1.2.1 operator=()

```
gsl::ccolumn_view & gsl::ccolumn_view::operator= (
            const cvector & v )
```

Assign data from complex vector to view.

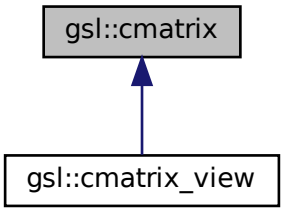
The documentation for this class was generated from the following files:

- /home/jspainhour/spheroidal_cpp/include/yawg/cvector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/cvector.cpp

7.2 gsl::cmatrix Class Reference

```
#include <cmatrix.h>
```

Inheritance diagram for gsl::cmatrix:



Public Member Functions

- cmatrix ()
  Construct empty matrix.
- cmatrix (size_t n, size_t m)
  Construct zero matrix of size n x m.
- cmatrix (FILE *in)
  Construct new gsl::cmatrix from MATLAB's .csv file format.
- cmatrix (const cvector &v)
  Construct new n x 1 gsl::cmatrix from a gsl::cvector.
- cmatrix (const cmatrix &M, size_t n, size_t m)
  Copy constructor creating n x m complex matrix.
- cmatrix (const cmatrix &M)
- cmatrix (cmatrix &&M)
- cmatrix (const matrix &M)
  Construct new gsl::cmatrix from gsl::matrix.
- cmatrix & operator= (const cmatrix &M)
- cmatrix & operator= (const matrix &M)

- cmatrix & operator= (cmatrix &&v)
- cmatrix & operator+= (const cmatrix &M)
- cmatrix & operator+= (const matrix &M)
- cmatrix & operator-= (const cmatrix &M)
- cmatrix & operator-= (const matrix &M)
- cmatrix & operator*= (complex z)
- cmatrix & operator*= (double x)
- cmatrix & operator/= (complex z)
- cmatrix & operator/= (double x)
- cmatrix operator- () const
- ~cmatrix ()
- complex_ref operator() (size_t i, size_t j)

  Return a reference to the element at position (i,j)
- void set (size_t i, size_t j, complex z)
- void set_col (size_t j, const cvector &v)
- void set_row (size_t i, const cvector &v)
- const complex_ref operator() (size_t i, size_t j) const

  Return a const reference to the element at position (i,j)
- complex get (size_t i, size_t j) const
- cvector get_col (size_t j) const
- cvector get_row (size_t i) const
- size_t size () const

  Size accessor.
- size_t nrows () const

  Number of rows accessor.
- size_t ncols () const

  Number of columns accessor.
- bool is_square () const
- gsl_matrix_complex * get () const

  Access the pointer to the underlying gsl_matrix_complex.
- void clear ()

  Clear the gsl::cmatrix, free underlying memory.
- void resize (size_t n, size_t m)

  Resize the gsl::cmatrix.
- cmatrix reshape (size_t n, size_t m) const

  Return a new n x m gsl::cmatrix with same elements.
- cmatrix & T ()

  Replace the complex matrix with its transpose.
- cmatrix & H ()

  Replace the complex matrix with its conjugate transpose.
- cmatrix & conj ()

  Return the conjugate of the complex matrix.
- void print (FILE *out=stdout) const

  Pretty-print the complex matrix to file stream.
- void print_csv (FILE *out=stdout) const

  Print the complex matrix to file stream in MATLAB's .csv format.
- void load_csv (FILE *in)

  Load the complex matrix from file stream in MATLAB's .csv format.
- cmatrix_view view () const
- cmatrix_view submatrix (size_t i, size_t j, size_t n, size_t m) const
- crow_view row (size_t i) const
- ccolumn_view column (size_t j) const

Protected Member Functions

- cmatrix (gsl_matrix_complex ∗gmat_other)

  Construct new gsl::matrix from gsl_matrix.
- void gfree ()

  Private function to free allocated memory.
- void galloc (size_t n, size_t m)

  Private function to (continuously) allocate memory.

Protected Attributes

- gsl_matrix_complex ∗ gmat

Friends

- cmatrix operator∗ (double a, const cmatrix &M)
- cmatrix operator∗ (double a, cmatrix &&M)
- cmatrix operator∗ (const cmatrix &M, double a)
- cmatrix operator∗ (cmatrix &&M, double a)
- cmatrix operator∗ (complex z, const cmatrix &M)
- cmatrix operator∗ (const cmatrix &M, complex z)
- cmatrix operator∗ (cmatrix &&M, complex z)
- cmatrix operator∗ (complex z, cmatrix &&M)
- cmatrix operator/ (double a, const cmatrix &M)
- cmatrix operator/ (double a, cmatrix &&M)
- cmatrix operator/ (const cmatrix &M, double a)
- cmatrix operator/ (cmatrix &&M, double a)
- cmatrix operator/ (complex z, const cmatrix &M)
- cmatrix operator/ (const cmatrix &M, complex z)
- cmatrix operator/ (cmatrix &&M, complex z)
- cmatrix operator/ (complex z, cmatrix &&M)
- cmatrix operator+ (const cmatrix &M1, const cmatrix &M2)
- cmatrix operator+ (cmatrix &&M1, const cmatrix &M2)
- cmatrix operator+ (const cmatrix &M1, cmatrix &&M2)
- cmatrix operator+ (cmatrix &&M1, cmatrix &&M2)
- cmatrix operator+ (const cmatrix &M1, const matrix &M2)
- cmatrix operator+ (cmatrix &&M1, const matrix &M2)
- cmatrix operator+ (const matrix &M1, const cmatrix &M2)
- cmatrix operator+ (const matrix &M1, cmatrix &&M2)
- cmatrix operator- (const cmatrix &M1, const cmatrix &M2)
- cmatrix operator- (cmatrix &&M1, const cmatrix &M2)
- cmatrix operator- (const cmatrix &M1, cmatrix &&M2)
- cmatrix operator- (cmatrix &&M1, cmatrix &&M2)
- cmatrix operator- (const cmatrix &M1, const matrix &M2)
- cmatrix operator- (cmatrix &&M1, const matrix &M2)
- cmatrix operator- (const matrix &M1, const cmatrix &M2)
- cmatrix operator- (const matrix &M1, cmatrix &&M2)
- cmatrix operator∗ (const cmatrix &A, const cmatrix &B)
- bool operator== (const cmatrix &M1, const cmatrix &M2)
- bool operator!= (const cmatrix &M1, const cmatrix &M2)
- bool operator== (const cmatrix &M1, const matrix &M2)
- bool operator== (const matrix &M1, const cmatrix &M2)
- bool operator!= (const cmatrix &M1, const matrix &M2)
- bool operator!= (const matrix &M1, const cmatrix &M2)
- cmatrix operator∗ (const cmatrix &A, const cmatrix &B)

7.2.1 Constructor & Destructor Documentation

7.2.1.1 cmatrix() [1/9]

```
gsl::cmatrix::cmatrix ( )
```

Construct empty matrix.

7.2.1.2 cmatrix() [2/9]

```
gsl::cmatrix::cmatrix (
            size_t n,
            size_t m )
```

Construct zero matrix of size n x m.

7.2.1.3 cmatrix() [3/9]

```
gsl::cmatrix::cmatrix (
            FILE * in )
```

Construct new gsl::cmatrix from MATLAB's .csv file format.

7.2.1.4 cmatrix() [4/9]

```
gsl::cmatrix::cmatrix (
            const cvector & v )
```

Construct new n x 1 gsl::cmatrix from a gsl::cvector.

7.2.1.5 cmatrix() [5/9]

```
gsl::cmatrix::cmatrix (
            const cmatrix & M,
            size_t n,
            size_t m )
```

Copy constructor creating n x m complex matrix.

Copy constructor creating n x m matrix.

Parameters

M	gsl::matrix to copy
n	Number of rows
m	Number of columns

7.2.1.6 cmatrix() [6/9]

```
gsl::cmatrix::cmatrix (
            const cmatrix & M )
```

7.2.1.7 cmatrix() [7/9]

```
gsl::cmatrix::cmatrix (
            gsl::cmatrix && gmat_other )
```

7.2.1.8 cmatrix() [8/9]

```
gsl::cmatrix::cmatrix (
            const matrix & M )
```

Construct new gsl::cmatrix from gsl::matrix.

Copy the values from a real matrix into a complex matrix, setting the imaginary part to zero.

7.2.1.9 ~cmatrix()

```
gsl::cmatrix::~cmatrix ( )
```

7.2.1.10 cmatrix() [9/9]

```
gsl::cmatrix::cmatrix (
            gsl_matrix_complex * gmat_other )  [protected]
```

Construct new gsl::matrix from gsl_matrix.

Construct new gsl::cvector from gsl_vector_complex's data.

7.2.2 Member Function Documentation

7.2.2.1 clear()

```
void gsl::cmatrix::clear ( )
```

Clear the gsl::cmatrix, free underlying memory.

CLear the gsl::cmatrix, free underlying memory.

7.2.2.2 column()

```
gsl::ccolumn_view gsl::cmatrix::column (
            size_t j ) const
```

7.2.2.3 conj()

```
gsl::cmatrix & gsl::cmatrix::conj ( )
```

Return the conjugate of the complex matrix.

Replace the matrix with its conjugate.

7.2.2.4 galloc()

```
void gsl::cmatrix::galloc (
            size_t n,
            size_t m )  [protected]
```

Private function to (continuously) allocate memory.

Note

This method allocates contiguous, zero-initialized memory. This is slightly slower than using gsl_matrix_↩ compllex_alloc, but allows for intuitive usage of row views.

7.2.2.5 get() [1/2]

```
gsl_matrix_complex* gsl::cmatrix::get ( ) const  [inline]
```

Access the pointer to the underlying gsl_matrix_complex.

7.2.2.6 get() [2/2]

```
gsl::complex gsl::cmatrix::get (
            size_t i,
            size_t j ) const
```

7.2.2.7 get_col()

```
gsl::cvector gsl::cmatrix::get_col (
            size_t j ) const
```

7.2.2.8 get_row()

```
gsl::cvector gsl::cmatrix::get_row (
            size_t i ) const
```

7.2.2.9 gfree()

```
void gsl::cmatrix::gfree ( )  [protected]
```

Private function to free allocated memory.

Free memory for underlying gsl_matrix_complex.

7.2.2.10 H()

```
gsl::cmatrix & gsl::cmatrix::H ( )
```

Replace the complex matrix with its conjugate transpose.

7.2.2.11 is_square()

```
bool gsl::cmatrix::is_square ( ) const
```

7.2.2.12 load_csv()

```
void gsl::cmatrix::load_csv (
            FILE * in )
```

Load the complex matrix from file stream in MATLAB's .csv format.

Parameters

in	File stream to load from

7.2.2.13 ncols()

```
size_t gsl::cmatrix::ncols ( ) const
```

Number of columns accessor.

7.2.2.14 nrows()

```
size_t gsl::cmatrix::nrows ( ) const
```

Number of rows accessor.

7.2.2.15 operator()() [1/2]

```
gsl::complex_ref gsl::cmatrix::operator() (
            size_t i,
            size_t j )
```

Return a reference to the element at position (i,j)

This function returns a complex_ref to the element at position (i,j) in the matrix. Allows setting

7.2.2.16 operator()() [2/2]

```
const gsl::complex_ref gsl::cmatrix::operator() (
            size_t i,
            size_t j ) const
```

Return a const reference to the element at position (i,j)

This function returns a constant complex_ref to the element at position (i,j) in the matrix. Allows getting.

7.2.2.17 operator*=() [1/2]

```
gsl::cmatrix & gsl::cmatrix::operator*= (
            complex z )
```

7.2.2.18 operator*=() [2/2]

```
gsl::cmatrix & gsl::cmatrix::operator*= (
            double x )
```

7.2.2.19 operator+=() [1/2]

```
gsl::cmatrix & gsl::cmatrix::operator+= (
            const cmatrix & M )
```

7.2.2.20 operator+=() [2/2]

```
gsl::cmatrix & gsl::cmatrix::operator+= (
            const matrix & M )
```

7.2.2.21 operator-()

```
gsl::cmatrix gsl::cmatrix::operator- ( ) const
```

7.2.2.22 operator-=() [1/2]

```
gsl::cmatrix & gsl::cmatrix::operator-= (
            const cmatrix & M )
```

7.2.2.23 operator-=() [2/2]

```
gsl::cmatrix & gsl::cmatrix::operator-= (
            const matrix & M )
```

7.2.2.24 operator/=() [1/2]

```
gsl::cmatrix & gsl::cmatrix::operator/= (
            complex z )
```

7.2.2.25 operator/=() [2/2]

```
gsl::cmatrix & gsl::cmatrix::operator/= (
            double x )
```

7.2.2.26 operator=() [1/3]

```
gsl::cmatrix & gsl::cmatrix::operator= (
            gsl::cmatrix && M )
```

7.2.2.27 operator=() [2/3]

```
gsl::cmatrix & gsl::cmatrix::operator= (
            const cmatrix & M )
```

7.2.2.28 operator=() [3/3]

```
gsl::cmatrix & gsl::cmatrix::operator= (
            const matrix & M )
```

7.2.2.29 print()

```
void gsl::cmatrix::print (
            FILE * out = stdout ) const
```

Pretty-print the complex matrix to file stream.

Parameters

out	File stream to print to

7.2.2.30 print_csv()

```
void gsl::cmatrix::print_csv (
            FILE * out = stdout ) const
```

Print the complex matrix to file stream in MATLAB's .csv format.

Parameters

out	File stream to print to

Note

This function uses a complex valued .csv format compatible with MATLAB's load/save functions, which has the following format 1.0000+2.0000i,2.0000+3.0000i,3.0000+4.0000i 2.0000+3.0000i,4.0000+5.0000i,6.↩0000+7.0000i

7.2.2.31 reshape()

```
gsl::cmatrix gsl::cmatrix::reshape (
            size_t n,
            size_t m ) const
```

Return a new n x m gsl::cmatrix with same elements.

Parameters

n	Number of rows
m	Number of columns

Returns

New gsl::matrix with same elements

7.2.2.32 resize()

```
void gsl::cmatrix::resize (
            size_t n,
            size_t m )
```

Resize the gsl::cmatrix.

Resize the gsl::cmatrix, setting elements to zero.

Parameters

n	Number of rows
m	Number of columns

Note

This function will always free and reallocate memory, setting the elements to zero.

7.2.2.33 row()

```
gsl::crow_view gsl::cmatrix::row (
            size_t i ) const
```

7.2.2.34 set()

```
void gsl::cmatrix::set (
            size_t i,
            size_t j,
            gsl::complex val )
```

7.2.2.35 set_col()

```
void gsl::cmatrix::set_col (
            size_t j,
            const cvector & v )
```

7.2.2.36 set_row()

```
void gsl::cmatrix::set_row (
            size_t i,
            const cvector & v )
```

7.2.2.37 size()

```
size_t gsl::cmatrix::size ( ) const
```

Size accessor.

7.2.2.38 submatrix()

```
gsl::cmatrix_view gsl::cmatrix::submatrix (
            size_t i,
            size_t j,
            size_t n,
            size_t m ) const
```

7.2.2.39 T()

```
gsl::cmatrix & gsl::cmatrix::T ( )
```

Replace the complex matrix with its transpose.

Note

If the matrix is not square, the transpose is not in-place

7.2.2.40 view()

```
gsl::cmatrix_view gsl::cmatrix::view ( ) const
```

7.2.3 Friends And Related Function Documentation

7.2.3.1 operator"!= [1/3]

```
bool operator!= (
            const cmatrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.2 operator"!= [2/3]

```
bool operator!= (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.2.3.3 operator"!= [3/3]

```
bool operator!= (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.4 operator* [1/10]

```
cmatrix operator* (
            cmatrix && M,
            complex z )  [friend]
```

7.2.3.5 operator* [2/10]

```
cmatrix operator* (
            cmatrix && M,
            double a )  [friend]
```

7.2.3.6 operator* [3/10]

```
cmatrix operator* (
            complex z,
            cmatrix && M )  [friend]
```

7.2.3.7 operator* [4/10]

```
cmatrix operator* (
            complex z,
            const cmatrix & M )  [friend]
```

7.2.3.8 operator* [5/10]

```
cmatrix operator* (
            const cmatrix & A,
            const cmatrix & B )  [friend]
```

7.2.3.9 operator* [6/10]

```
cmatrix operator* (
            const cmatrix & A,
            const cmatrix & B )  [friend]
```

7.2.3.10 operator* [7/10]

```
cmatrix operator* (
            const cmatrix & M,
            complex z )  [friend]
```

7.2.3.11 operator* [8/10]

```
cmatrix operator* (
            const cmatrix & M,
            double a )  [friend]
```

7.2.3.12 operator* [9/10]

```
cmatrix operator* (
            double a,
            cmatrix && M )  [friend]
```

7.2.3.13 operator* [10/10]

```
cmatrix operator* (
            double a,
            const cmatrix & M )  [friend]
```

7.2.3.14 operator+ [1/8]

```
cmatrix operator+ (
            cmatrix && M1,
            cmatrix && M2 )  [friend]
```

7.2.3.15 operator+ [2/8]

```
cmatrix operator+ (
            cmatrix && M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.16 operator+ [3/8]

```
cmatrix operator+ (
            cmatrix && M1,
            const matrix & M2 )  [friend]
```

7.2.3.17 operator+ [4/8]

```
cmatrix operator+ (
            const cmatrix & M1,
            cmatrix && M2 )  [friend]
```

7.2.3.18 operator+ [5/8]

```
cmatrix operator+ (
            const cmatrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.19 operator+ [6/8]

```
cmatrix operator+ (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.2.3.20 operator+ [7/8]

```
cmatrix operator+ (
            const matrix & M1,
            cmatrix && M2 )  [friend]
```

7.2.3.21 operator+ [8/8]

```
cmatrix operator+ (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.22 operator- [1/8]

```
cmatrix operator- (
            cmatrix && M1,
            cmatrix && M2 )  [friend]
```

7.2.3.23 operator- [2/8]

```
cmatrix operator- (
            cmatrix && M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.24 operator- [3/8]

```
cmatrix operator- (
            cmatrix && M1,
            const matrix & M2 )  [friend]
```

7.2.3.25 operator- [4/8]

```
cmatrix operator- (
            const cmatrix & M1,
            cmatrix && M2 )  [friend]
```

7.2.3.26 operator- [5/8]

```
cmatrix operator- (
            const cmatrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.27 operator- [6/8]

```
cmatrix operator- (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.2.3.28 operator- [7/8]

```
cmatrix operator- (
            const matrix & M1,
            cmatrix && M2 )  [friend]
```

7.2.3.29 operator- [8/8]

```
cmatrix operator- (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.30 operator/ [1/8]

```
cmatrix operator/ (
            cmatrix && M,
            complex z )  [friend]
```

7.2.3.31 operator/ [2/8]

```
cmatrix operator/ (
            cmatrix && M,
            double a )  [friend]
```

7.2.3.32 operator/ [3/8]

```
cmatrix operator/ (
            complex z,
            cmatrix && M )  [friend]
```

7.2.3.33 operator/ [4/8]

```
cmatrix operator/ (
            complex z,
            const cmatrix & M )  [friend]
```

7.2.3.34 operator/ [5/8]

```
cmatrix operator/ (
            const cmatrix & M,
            complex z )  [friend]
```

7.2.3.35 operator/ [6/8]

```
cmatrix operator/ (
            const cmatrix & M,
            double a )  [friend]
```

7.2.3.36 operator/ [7/8]

```
cmatrix operator/ (
            double a,
            cmatrix && M )  [friend]
```

7.2.3.37 operator/ [8/8]

```
cmatrix operator/ (
            double a,
            const cmatrix & M )  [friend]
```

7.2.3.38 operator== [1/3]

```
bool operator== (
            const cmatrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.3.39 operator== [2/3]

```
bool operator== (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.2.3.40 operator== [3/3]

```
bool operator== (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.2.4 Member Data Documentation

7.2.4.1 gmat

```
gsl_matrix_complex* gsl::cmatrix::gmat  [protected]
```

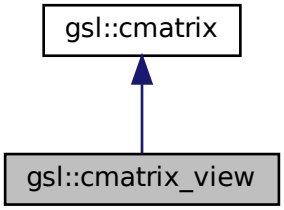
The documentation for this class was generated from the following files:

- /home/jspainhour/spheroidal_cpp/include/yawg/cmatrix.h
- /home/jspainhour/spheroidal_cpp/src_yawg/cmatrix.cpp

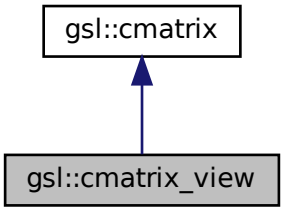
7.3 gsl::cmatrix_view Class Reference

```
#include <cmatrix.h>
```

Inheritance diagram for gsl::cmatrix_view:



Collaboration diagram for gsl::cmatrix_view:



Public Member Functions

- cmatrix_view (gsl_matrix_complex *gvec_other)

  Constructor for vector_view pointing to data at gvec_other.
- ~cmatrix_view ()
- cmatrix_view & operator= (const cmatrix &M)

  Assign data from matrix to view.
- void clear ()
- void resize (size_t n, size_t m)

Additional Inherited Members

7.3.1 Constructor & Destructor Documentation

7.3.1.1 cmatrix_view()

```
gsl::cmatrix_view::cmatrix_view (
            gsl_matrix_complex * gvec_other )
```

Constructor for vector_view pointing to data at gvec_other.

7.3.1.2 ~cmatrix_view()

```
gsl::cmatrix_view::~cmatrix_view ( )
```

7.3.2 Member Function Documentation

7.3.2.1 clear()

```
void gsl::cmatrix_view::clear ( )
```

7.3.2.2 operator=()

```
gsl::cmatrix_view & gsl::cmatrix_view::operator= (
            const cmatrix & M )
```

Assign data from matrix to view.

7.3.2.3 resize()

```
void gsl::cmatrix_view::resize (
            size_t n,
            size_t m )
```

The documentation for this class was generated from the following files:

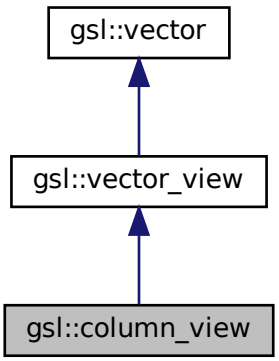
- /home/jspainhour/spheroidal_cpp/include/yawg/cmatrix.h
- /home/jspainhour/spheroidal_cpp/src_yawg/cmatrix.cpp

7.4 gsl::column_view Class Reference

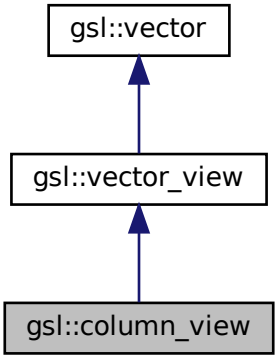
A subclass of vector_view for non-stride-1 vectors.

`#include <vector.h>`

Inheritance diagram for gsl::column_view:



Collaboration diagram for gsl::column_view:



Public Member Functions

- column_view (gsl_vector *gvec_other)

  Construct column_view from existing vector view.
- column_view & operator= (const vector &v)

  Assign data from vector to view.

Additional Inherited Members

7.4.1 Detailed Description

A subclass of vector_view for non-stride-1 vectors.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 column_view()

```
gsl::column_view::column_view (
            gsl_vector * gvec_other )  [inline]
```

Construct column_view from existing vector view.

7.4.3 Member Function Documentation

7.4.3.1 operator=()

```
gsl::column_view & gsl::column_view::operator= (
            const vector & v )
```

Assign data from vector to view.

The documentation for this class was generated from the following files:

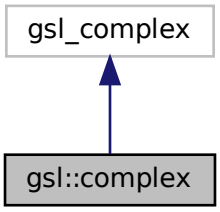
- /home/jspainhour/spheroidal_cpp/include/yawg/vector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/vector.cpp

7.5 gsl::complex Class Reference

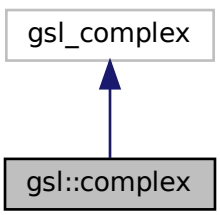
Wrapper class for gsl_complex structs.

```
#include <complex.h>
```

Inheritance diagram for gsl::complex:



Collaboration diagram for gsl::complex:



Public Member Functions

- complex ()
- complex (double x)
- complex (double re, double im)
- complex (gsl_complex z)
- double real () const
- double imag () const
- double abs () const
- double abs2 () const
- double arg () const
- complex & operator+= (complex gsl_complex_other)
- complex & operator-= (complex gsl_complex_other)
- complex & operator*= (complex gsl_complex_other)
- complex & operator/= (complex gsl_complex_other)
- complex & operator+= (double x)
- complex & operator-= (double x)
- complex & operator*= (double x)
- complex & operator/= (double x)
- complex & operator= (complex gsl_complex_other)
- void set (double re, double im)
- void set (complex z)
- complex operator- () const
- void print () const

Friends

- class complex_ref
- complex operator+ (const complex &a, const complex &b)
- complex operator- (const complex &a, const complex &b)
- complex operator* (const complex &a, const complex &b)
- complex operator/ (const complex &a, const complex &b)
- bool operator== (const complex &a, const complex &b)
- bool operator!= (const complex &a, const complex &b)
- complex operator+ (const complex &a, double b)
- complex operator- (const complex &a, double b)
- complex operator* (const complex &a, double b)
- complex operator/ (const complex &a, double b)
- bool operator== (const complex &a, double b)
- bool operator!= (const complex &a, double b)
- complex operator+ (double a, const complex &b)
- complex operator- (double a, const complex &b)
- complex operator* (double a, const complex &b)
- complex operator/ (double a, const complex &b)
- bool operator== (double a, const complex &b)
- bool operator!= (double a, const complex &b)

7.5.1 Detailed Description

Wrapper class for gsl_complex structs.

Inherits `double dat[2]` from gsl_complex and provides a number of convenience functions.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 complex() [1/4]

```
gsl::complex::complex ( )  [inline]
```

7.5.2.2 complex() [2/4]

```
gsl::complex::complex (
            double x )  [inline]
```

7.5.2.3 complex() [3/4]

```
gsl::complex::complex (
            double re,
            double im )  [inline]
```

7.5.2.4 complex() [4/4]

```
gsl::complex::complex (
            gsl_complex z )  [inline]
```

7.5.3 Member Function Documentation

7.5.3.1 abs()

```
double gsl::complex::abs ( ) const  [inline]
```

7.5.3.2 abs2()

```
double gsl::complex::abs2 ( ) const  [inline]
```

7.5.3.3 arg()

```
double gsl::complex::arg ( ) const  [inline]
```

7.5.3.4 imag()

```
double gsl::complex::imag ( ) const  [inline]
```

7.5.3.5 operator*=() [1/2]

```
gsl::complex & gsl::complex::operator*= (
            complex gsl_complex_other )
```

7.5.3.6 operator*=() [2/2]

```
gsl::complex & gsl::complex::operator*= (
            double x )
```

7.5.3.7 operator+=() [1/2]

```
gsl::complex & gsl::complex::operator+= (
            complex gsl_complex_other )
```

7.5.3.8 operator+=() [2/2]

```
gsl::complex & gsl::complex::operator+= (
            double x )
```

7.5.3.9 operator-()

```
complex gsl::complex::operator- ( ) const  [inline]
```

7.5.3.10 operator-=() [1/2]

```
gsl::complex & gsl::complex::operator-= (
            complex gsl_complex_other )
```

7.5.3.11 operator-=() [2/2]

```
gsl::complex & gsl::complex::operator-= (
            double x )
```

7.5.3.12 operator/=() [1/2]

```
gsl::complex & gsl::complex::operator/= (
            complex gsl_complex_other )
```

7.5.3.13 operator/=() [2/2]

```
gsl::complex & gsl::complex::operator/= (
            double x )
```

7.5.3.14 operator=()

```
gsl::complex & gsl::complex::operator= (
            gsl::complex z )
```

7.5.3.15 print()

```
void gsl::complex::print ( ) const
```

7.5.3.16 real()

```
double gsl::complex::real ( ) const  [inline]
```

7.5.3.17 set() [1/2]

```
void gsl::complex::set (
            complex z )  [inline]
```

7.5.3.18 set() [2/2]

```
void gsl::complex::set (
            double re,
            double im )  [inline]
```

7.5.4 Friends And Related Function Documentation

7.5.4.1 complex_ref

```
friend class complex_ref  [friend]
```

7.5.4.2 operator"!= [1/3]

```
bool operator!= (
            const complex & a,
            const complex & b )  [friend]
```

7.5.4.3 operator"!= [2/3]

```
bool operator!= (
            const complex & a,
            double b )  [friend]
```

7.5.4.4 operator"!= [3/3]

```
bool operator!= (
            double a,
            const complex & b )  [friend]
```

7.5.4.5 operator* [1/3]

```
complex operator* (
            const complex & a,
            const complex & b )  [friend]
```

7.5.4.6 operator* [2/3]

```
complex operator* (
            const complex & a,
            double b )  [friend]
```

7.5.4.7 operator* [3/3]

```
complex operator* (
            double a,
            const complex & b )  [friend]
```

7.5.4.8 operator+ [1/3]

```
complex operator+ (
            const complex & a,
            const complex & b )  [friend]
```

7.5.4.9 operator+ [2/3]

```
complex operator+ (
            const complex & a,
            double b )  [friend]
```

7.5.4.10 operator+ [3/3]

```
complex operator+ (
            double a,
            const complex & b )  [friend]
```

7.5.4.11 operator- [1/3]

```
complex operator- (
            const complex & a,
            const complex & b )  [friend]
```

7.5.4.12 operator- [2/3]

```
complex operator- (
            const complex & a,
            double b )  [friend]
```

7.5.4.13 operator- [3/3]

```
complex operator- (
            double a,
            const complex & b )  [friend]
```

7.5.4.14 operator/ [1/3]

```
complex operator/ (
            const complex & a,
            const complex & b )  [friend]
```

7.5.4.15 operator/ [2/3]

```
complex operator/ (
            const complex & a,
            double b )  [friend]
```

7.5.4.16 operator/ [3/3]

```
complex operator/ (
            double a,
            const complex & b )  [friend]
```

7.5.4.17 operator== [1/3]

```
bool operator== (
            const complex & a,
            const complex & b )  [friend]
```

7.5.4.18 operator== [2/3]

```
bool operator== (
            const complex & a,
            double b )  [friend]
```

7.5.4.19 operator== [3/3]

```
bool operator== (
            double a,
            const complex & b )  [friend]
```

The documentation for this class was generated from the following files:

- /home/jspainhour/spheroidal_cpp/include/yawg/complex.h
- /home/jspainhour/spheroidal_cpp/src_yawg/complex.cpp

7.6 gsl::complex_ref Class Reference

Stores a refernce to a gsl::complex object.

`#include <complex.h>`

Public Member Functions

- operator complex () const

  "Dereferences" a complex_ref into independent gsl::complex object
- complex operator∗ () const

  Dereferences a complex_ref into independent gsl_complex object.
- complex_ref (complex_ref &z)

  Constructs a reference to another compelx_ref object.
- complex_ref (gsl_complex ∗z)

  Constructs a reference to a gsl_complex struct.
- complex_ref (complex &z)

  Constructs a reference to a gsl::complex object.
- complex_ref & operator+= (const complex &gsl_complex_other)
- complex_ref & operator-= (const complex &gsl_complex_other)
- complex_ref & operator∗= (const complex &gsl_complex_other)

- complex_ref & operator/= (const complex &gsl_complex_other)
- complex_ref & operator+= (double x)
- complex_ref & operator-= (double x)
- complex_ref & operator∗= (double x)
- complex_ref & operator/= (double x)
- complex_ref & operator= (const complex &z)

  Assigns values of gsl::complex object to the reference.
- complex_ref & operator= (const complex_ref &z)

  Assigns values of one complex_ref object to another.
- double real () const
- double imag () const

Protected Member Functions

- complex_ref ()

Protected Attributes

- double ∗ dat

Friends

- class complex
- complex operator+ (const complex_ref &a, const complex_ref &b)
- complex operator- (const complex_ref &a, const complex_ref &b)
- complex operator∗ (const complex_ref &a, const complex_ref &b)
- complex operator/ (const complex_ref &a, const complex_ref &b)
- bool operator== (const complex_ref &a, const complex_ref &b)
- bool operator!= (const complex_ref &a, const complex_ref &b)
- complex operator+ (const complex_ref &a, double b)
- complex operator- (const complex_ref &a, double b)
- complex operator∗ (const complex_ref &a, double b)
- complex operator/ (const complex_ref &a, double b)
- bool operator== (const complex_ref &a, double b)
- bool operator!= (const complex_ref &a, double b)
- complex operator+ (double a, const complex_ref &b)
- complex operator- (double a, const complex_ref &b)
- complex operator∗ (double a, const complex_ref &b)
- complex operator/ (double a, const complex_ref &b)
- bool operator== (double a, const complex_ref &b)
- bool operator!= (double a, const complex_ref &b)

7.6.1 Detailed Description

Stores a refernce to a gsl::complex object.

This class is necessary to communicate between gsl_complex and gsl::complex so that overloads of () work gsl::cvector and gsl::cmatrix.

Implementation heavily inspired by ccgsl (https://ccgsl.sourceforge.net/)

7.6.2 Constructor & Destructor Documentation

7.6.2.1 complex_ref() [1/4]

```
gsl::complex_ref::complex_ref ( )  [inline], [protected]
```

7.6.2.2 complex_ref() [2/4]

```
gsl::complex_ref::complex_ref (
            complex_ref & z )  [inline]
```

Constructs a reference to another compelx_ref object.

7.6.2.3 complex_ref() [3/4]

```
gsl::complex_ref::complex_ref (
            gsl_complex * z )  [inline]
```

Constructs a reference to a gsl_complex struct.

7.6.2.4 complex_ref() [4/4]

```
gsl::complex_ref::complex_ref (
            complex & z )  [inline]
```

Constructs a reference to a gsl::complex object.

7.6.3 Member Function Documentation

7.6.3.1 imag()

```
double gsl::complex_ref::imag ( ) const  [inline]
```

7.6.3.2 operator complex()

```
gsl::complex_ref::operator complex ( ) const  [inline]
```

"Dereferences" a complex_ref into independent gsl::complex object

7.6.3.3 operator∗()

```
complex gsl::complex_ref::operator* ( ) const  [inline]
```

Dereferences a complex_ref into independent gsl_complex object.

7.6.3.4 operator∗=() [1/2]

```
gsl::complex_ref & gsl::complex_ref::operator*= (
            const complex & gsl_complex_other )
```

7.6.3.5 operator∗=() [2/2]

```
gsl::complex_ref & gsl::complex_ref::operator*= (
            double x )
```

7.6.3.6 operator+=() [1/2]

```
gsl::complex_ref & gsl::complex_ref::operator+= (
            const complex & gsl_complex_other )
```

7.6.3.7 operator+=() [2/2]

```
gsl::complex_ref & gsl::complex_ref::operator+= (
            double x )
```

7.6.3.8 operator-=() [1/2]

```
gsl::complex_ref & gsl::complex_ref::operator-= (
            const complex & gsl_complex_other )
```

7.6.3.9 operator-=() [2/2]

```
gsl::complex_ref & gsl::complex_ref::operator-= (
            double x )
```

7.6.3.10 operator/=() [1/2]

```
gsl::complex_ref & gsl::complex_ref::operator/= (
            const complex & gsl_complex_other )
```

7.6.3.11 operator/=() [2/2]

```
gsl::complex_ref & gsl::complex_ref::operator/= (
            double x )
```

7.6.3.12 operator=() [1/2]

```
complex_ref& gsl::complex_ref::operator= (
            const complex & z )  [inline]
```

Assigns values of gsl::complex object to the reference.

7.6.3.13 operator=() [2/2]

```
complex_ref& gsl::complex_ref::operator= (
            const complex_ref & z )  [inline]
```

Assigns values of one complex_ref object to another.

Note

This is an alternative to the default assignment operator, which does not work for unknown reasons.

7.6.3.14 real()

```
double gsl::complex_ref::real ( ) const  [inline]
```

7.6.4 Friends And Related Function Documentation

7.6.4.1 complex

```
friend class complex  [friend]
```

7.6.4.2 operator"!= [1/3]

```
bool operator!= (
            const complex_ref & a,
            const complex_ref & b )  [friend]
```

7.6.4.3 operator"!= [2/3]

```
bool operator!= (
            const complex_ref & a,
            double b )  [friend]
```

7.6.4.4 operator"!= [3/3]

```
bool operator!= (
            double a,
            const complex_ref & b )  [friend]
```

7.6.4.5 operator* [1/3]

```
complex operator* (
            const complex_ref & a,
            const complex_ref & b )  [friend]
```

7.6.4.6 operator* [2/3]

```
complex operator* (
            const complex_ref & a,
            double b )  [friend]
```

7.6.4.7 operator* [3/3]

```
complex operator* (
            double a,
            const complex_ref & b )  [friend]
```

7.6.4.8 operator+ [1/3]

```
complex operator+ (
            const complex_ref & a,
            const complex_ref & b )  [friend]
```

7.6.4.9 operator+ [2/3]

```
complex operator+ (
            const complex_ref & a,
            double b )  [friend]
```

7.6.4.10 operator+ [3/3]

```
complex operator+ (
            double a,
            const complex_ref & b )  [friend]
```

7.6.4.11 operator- [1/3]

```
complex operator- (
            const complex_ref & a,
            const complex_ref & b )  [friend]
```

7.6.4.12 operator- [2/3]

```
complex operator- (
            const complex_ref & a,
            double b )  [friend]
```

7.6.4.13 operator- [3/3]

```
complex operator- (
            double a,
            const complex_ref & b )  [friend]
```

7.6.4.14 operator/ [1/3]

```
complex operator/ (
            const complex_ref & a,
            const complex_ref & b )  [friend]
```

7.6.4.15 operator/ [2/3]

```
complex operator/ (
            const complex_ref & a,
            double b )  [friend]
```

7.6.4.16 operator/ [3/3]

```
complex operator/ (
            double a,
            const complex_ref & b )  [friend]
```

7.6.4.17 operator== [1/3]

```
bool operator== (
            const complex_ref & a,
            const complex_ref & b )  [friend]
```

7.6.4.18 operator== [2/3]

```
bool operator== (
            const complex_ref & a,
            double b )  [friend]
```

7.6.4.19 operator== [3/3]

```
bool operator== (
            double a,
            const complex_ref & b )  [friend]
```

7.6.5 Member Data Documentation

7.6.5.1 dat

```
double* gsl::complex_ref::dat  [protected]
```

The documentation for this class was generated from the following files:

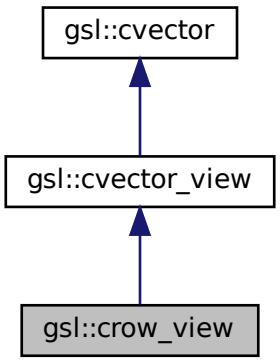
- /home/jspainhour/spheroidal_cpp/include/yawg/complex.h
- /home/jspainhour/spheroidal_cpp/src_yawg/complex.cpp

7.7 gsl::crow_view Class Reference

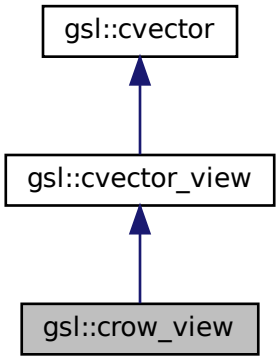
A subclass of cvector_view for stride-1 cvectors.

```
#include <cvector.h>
```

Inheritance diagram for gsl::crow_view:



Collaboration diagram for gsl::crow_view:



Public Member Functions

- crow_view (gsl_vector_complex ∗gvec_other)

  Construct row_view from existing vector view, checking that stride is 1.
- crow_view & operator= (const cvector &v)

  Assign data from complex vector to view.
- cmatrix_view reshape (size_t n, size_t m) const

  Return a matrix view out of the elements of the row.

Additional Inherited Members

7.7.1 Detailed Description

A subclass of cvector_view for stride-1 cvectors.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 crow_view()

```
gsl::crow_view::crow_view (
            gsl_vector_complex * gvec_other )  [inline]
```

Construct row_view from existing vector view, checking that stride is 1.

7.7.3 Member Function Documentation

7.7.3.1 operator=()

```
gsl::crow_view & gsl::crow_view::operator= (
            const cvector & v )
```

Assign data from complex vector to view.

7.7.3.2 reshape()

```
gsl::cmatrix_view gsl::crow_view::reshape (
            size_t n,
            size_t m ) const
```

Return a matrix view out of the elements of the row.

The documentation for this class was generated from the following files:

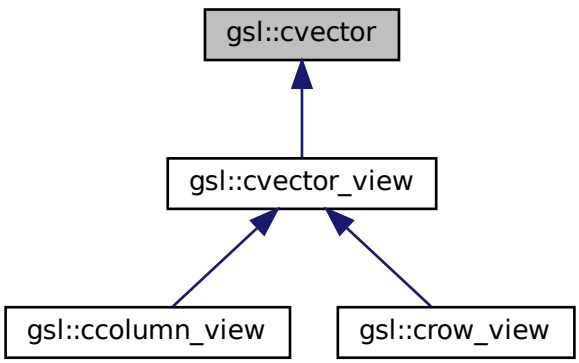
- /home/jspainhour/spheroidal_cpp/include/yawg/cvector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/cvector.cpp

7.8 gsl::cvector Class Reference

A wrapper class for gsl_vector_complex.

```
#include <cvector.h>
```

Inheritance diagram for gsl::cvector:



Public Member Functions

- cvector ()

  Construct empty vector.
- cvector (size_t n)

  Construct zero vector of size n.
- cvector (gsl_vector_complex ∗gvec_other)

  Construct new gsl::cvector from gsl_vector_complex's data.
- cvector (const cvector &gvec_other)
- cvector (cvector &&gvec_other)
- cvector (const vector &vec)

  Construct new gsl::cvector from gsl::vector.
- cvector & operator= (const cvector &v)
- cvector & operator= (const vector &v)
- cvector & operator= (cvector &&gvec_other)
- cvector & operator+= (const cvector &v)
- cvector & operator+= (const vector &v)
- cvector & operator-= (const cvector &v)
- cvector & operator-= (const vector &v)
- cvector & operator∗= (complex a)
- cvector & operator/= (complex a)
- cvector & operator∗= (double a)
- cvector & operator/= (double a)
- cvector operator- () const
- ∼cvector ()
- complex_ref operator() (size_t i)

  Return a reference to the element at position (i,j)
- void set (size_t i, complex z)
- const complex_ref operator() (size_t i) const

  Return a const reference to the element at position (i,j)
- complex get (size_t i) const
- size_t size () const
- gsl_vector_complex ∗ get () const

  Access the pointer to the underlying gsl_vector_complex.
- void resize (size_t n)

  Resize the gsl::cvector, setting elements to zero.
- void clear ()

  Clear the gsl::cvector, free underlying memory.
- void print (FILE ∗out=stdout) const

  Pretty-print the complex vector to file stream.
- double norm () const

  Return the 2-norm of the vector.
- cvector_view view () const
- cvector_view subvector (size_t offset, size_t n) const

Protected Member Functions

- cvector (const gsl_vector_complex ∗gvec_other)

  Construct new gsl::cvector from gsl_vector_complex.
- void gfree ()

  Private function to free allocated memory.
- void galloc (size_t n)

  Private function to (continuously) allocate memory.

Protected Attributes

- gsl_vector_complex * gvec

Friends

- cvector operator* (complex a, const cvector &v)
- cvector operator* (complex a, cvector &&v)
- cvector operator* (const cvector &v, complex a)
- cvector operator* (cvector &&v, complex a)
- cvector operator* (const cvector &v, double x)
- cvector operator* (double x, const cvector &v)
- cvector operator* (cvector &&v, double x)
- cvector operator* (double x, cvector &&v)
- cvector operator/ (complex a, const cvector &v)
- cvector operator/ (complex a, cvector &&v)
- cvector operator/ (const cvector &v, complex a)
- cvector operator/ (cvector &&v, complex a)
- cvector operator/ (const cvector &v, double x)
- cvector operator/ (double x, const cvector &v)
- cvector operator/ (cvector &&v, double x)
- cvector operator/ (double x, cvector &&v)
- cvector operator+ (const cvector &v1, const cvector &v2)
- cvector operator+ (cvector &&v1, const cvector &v2)
- cvector operator+ (const cvector &v1, cvector &&v2)
- cvector operator+ (cvector &&v1, cvector &&v2)
- cvector operator+ (const vector &v1, const cvector &v2)
- cvector operator+ (const vector &v1, cvector &&v2)
- cvector operator+ (const cvector &v1, const vector &v2)
- cvector operator+ (cvector &&v1, const vector &v2)
- cvector operator- (const cvector &v1, const cvector &v2)
- cvector operator- (cvector &&v1, const cvector &v2)
- cvector operator- (const cvector &v1, cvector &&v2)
- cvector operator- (cvector &&v1, cvector &&v2)
- cvector operator- (const vector &v1, const cvector &v2)
- cvector operator- (const vector &v1, cvector &&v2)
- cvector operator- (const cvector &v1, const vector &v2)
- cvector operator- (cvector &&v1, const vector &v2)
- bool operator== (const cvector &v1, const cvector &v2)
- bool operator!= (const cvector &v1, const cvector &v2)
- bool operator== (const vector &v1, const cvector &v2)
- bool operator== (const cvector &v1, const vector &v2)
- bool operator!= (const vector &v1, const cvector &v2)
- bool operator!= (const cvector &v1, const vector &v2)
- cvector operator* (const cmatrix &M, const cvector &v)

7.8.1 Detailed Description

A wrapper class for gsl_vector_complex.

Stores and operates on a pointer to a gsl_vector_complex.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 cvector() [1/7]

```
gsl::cvector::cvector ( )
```

Construct empty vector.

Construct empty cvector.

7.8.2.2 cvector() [2/7]

```
gsl::cvector::cvector (
            size_t n )  [explicit]
```

Construct zero vector of size n.

Construct zero cvector of size n.

7.8.2.3 cvector() [3/7]

```
gsl::cvector::cvector (
            gsl_vector_complex * gvec_other )
```

Construct new gsl::cvector from gsl_vector_complex's data.

7.8.2.4 cvector() [4/7]

```
gsl::cvector::cvector (
            const cvector & gvec_other )
```

7.8.2.5 cvector() [5/7]

```
gsl::cvector::cvector (
            gsl::cvector && gvec_other )
```

7.8.2.6 cvector() [6/7]

```
gsl::cvector::cvector (
            const vector & vec )
```

Construct new gsl::cvector from gsl::vector.

Construct new gsl::cmatrix from gsl::matrix.

Copy the values from a real matrix into a complex matrix, setting the imaginary part to zero.

7.8.2.7 ~cvector()

```
gsl::cvector::~cvector ( )
```

7.8.2.8 cvector() [7/7]

```
gsl::cvector::cvector (
            const gsl_vector_complex * gvec_other )  [protected]
```

Construct new gsl::cvector from gsl_vector_complex.

7.8.3 Member Function Documentation

7.8.3.1 clear()

```
void gsl::cvector::clear ( )
```

Clear the gsl::cvector, free underlying memory.

Clear the gsl::vector, free underlying memory.

7.8.3.2 galloc()

```
void gsl::cvector::galloc (
            size_t n )  [protected]
```

Private function to (continuously) allocate memory.

Note

This method allocates contiguous, zero-initialized memory. This is slightly slower than using gsl_vector_↩ complex_alloc, but allows for intuitive usage of row views.

7.8.3.3 get() [1/2]

```
gsl_vector_complex* gsl::cvector::get ( ) const  [inline]
```

Access the pointer to the underlying gsl_vector_complex.

7.8.3.4 get() [2/2]

```
gsl::complex gsl::cvector::get (
            size_t i ) const
```

7.8.3.5 gfree()

```
void gsl::cvector::gfree ( )  [protected]
```

Private function to free allocated memory.

7.8.3.6 norm()

```
double gsl::cvector::norm ( ) const  [inline]
```

Return the 2-norm of the vector.

7.8.3.7 operator()() [1/2]

```
gsl::complex_ref gsl::cvector::operator() (
            size_t i )
```

Return a reference to the element at position (i,j)

This function returns a complex_ref to the element at position (i,j) in the matrix. Allows setting.

7.8.3.8 operator()() [2/2]

```
const gsl::complex_ref gsl::cvector::operator() (
            size_t i ) const
```

Return a const reference to the element at position (i,j)

This function returns a constant complex_ref to the element at position (i,j) in the matrix. Allows getting.

7.8.3.9 operator*=() [1/2]

```
gsl::cvector & gsl::cvector::operator*= (
            gsl::complex z )
```

7.8.3.10 operator*=() [2/2]

```
gsl::cvector & gsl::cvector::operator*= (
            double a )
```

7.8.3.11 operator+=() [1/2]

```
gsl::cvector & gsl::cvector::operator+= (
            const cvector & v )
```

7.8.3.12 operator+=() [2/2]

```
gsl::cvector & gsl::cvector::operator+= (
            const vector & v )
```

7.8.3.13 operator-()

```
gsl::cvector gsl::cvector::operator- ( ) const
```

7.8.3.14 operator-=() [1/2]

```
gsl::cvector & gsl::cvector::operator-= (
            const cvector & v )
```

7.8.3.15 operator-=() [2/2]

```
gsl::cvector & gsl::cvector::operator-= (
            const vector & v )
```

7.8.3.16 operator/=() [1/2]

```
gsl::cvector & gsl::cvector::operator/= (
            gsl::complex z )
```

7.8.3.17 operator/=() [2/2]

```
gsl::cvector & gsl::cvector::operator/= (
            double a )
```

7.8.3.18 operator=() [1/3]

```
gsl::cvector & gsl::cvector::operator= (
            const cvector & v )
```

7.8.3.19 operator=() [2/3]

```
gsl::cvector & gsl::cvector::operator= (
            const vector & v )
```

7.8.3.20 operator=() [3/3]

```
gsl::cvector & gsl::cvector::operator= (
            gsl::cvector && v )
```

7.8.3.21 print()

```
void gsl::cvector::print (
            FILE * out = stdout ) const
```

Pretty-print the complex vector to file stream.

Pretty-print the vector to file stream.

Parameters

out	File stream to print to

7.8.3.22 resize()

```
void gsl::cvector::resize (
            size_t n )
```

Resize the gsl::cvector, setting elements to zero.

Parameters

n	Number of elements

Note

This function will always free and reallocate memory, setting the elements to zero.

7.8.3.23 set()

```
void gsl::cvector::set (
            size_t i,
            gsl::complex val )
```

7.8.3.24 size()

```
size_t gsl::cvector::size ( ) const
```

7.8.3.25 subvector()

```
gsl::cvector_view gsl::cvector::subvector (
            size_t offset,
            size_t n ) const
```

7.8.3.26 view()

```
gsl::cvector_view gsl::cvector::view ( ) const
```

7.8.4 Friends And Related Function Documentation

7.8.4.1 operator"!= [1/3]

```
bool operator!= (
            const cvector & v1,
            const cvector & v2 )  [friend]
```

7.8.4.2 operator"!= [2/3]

```
bool operator!= (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.8.4.3 operator"!= [3/3]

```
bool operator!= (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.8.4.4 operator* [1/9]

```
cvector operator* (
            complex a,
            const cvector & v )  [friend]
```

7.8.4.5 operator* [2/9]

```
cvector operator* (
            complex a,
            cvector && v )  [friend]
```

7.8.4.6 operator* [3/9]

```
cvector operator* (
            const cmatrix & M,
            const cvector & v )  [friend]
```

7.8.4.7 operator* [4/9]

```
cvector operator* (
            const cvector & v,
            complex a )  [friend]
```

7.8.4.8 operator* [5/9]

```
cvector operator* (
            const cvector & v,
            double x )  [friend]
```

7.8.4.9 operator* [6/9]

```
cvector operator* (
            cvector && v,
            complex a )  [friend]
```

7.8.4.10 operator* [7/9]

```
cvector operator* (
            cvector && v,
            double x )  [friend]
```

7.8.4.11 operator* [8/9]

```
cvector operator* (
            double x,
            const cvector & v )  [friend]
```

7.8.4.12 operator* [9/9]

```
cvector operator* (
            double x,
            cvector && v )  [friend]
```

7.8.4.13 operator+ [1/8]

```
cvector operator+ (
            const cvector & v1,
            const cvector & v2 )  [friend]
```

7.8.4.14 operator+ [2/8]

```
cvector operator+ (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.8.4.15 operator+ [3/8]

```
cvector operator+ (
            const cvector & v1,
            cvector && v2 )  [friend]
```

7.8.4.16 operator+ [4/8]

```
cvector operator+ (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.8.4.17 operator+ [5/8]

```
cvector operator+ (
            const vector & v1,
            cvector && v2 )  [friend]
```

7.8.4.18 operator+ [6/8]

```
cvector operator+ (
            cvector && v1,
            const cvector & v2 )  [friend]
```

7.8.4.19 operator+ [7/8]

```
cvector operator+ (
            cvector && v1,
            const vector & v2 )  [friend]
```

7.8.4.20 operator+ [8/8]

```
cvector operator+ (
            cvector && v1,
            cvector && v2 )  [friend]
```

7.8.4.21 operator- [1/8]

```
cvector operator- (
            const cvector & v1,
            const cvector & v2 )  [friend]
```

7.8.4.22 operator- [2/8]

```
cvector operator- (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.8.4.23 operator- [3/8]

```
cvector operator- (
            const cvector & v1,
            cvector && v2 )  [friend]
```

7.8.4.24 operator- [4/8]

```
cvector operator- (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.8.4.25 operator- [5/8]

```
cvector operator- (
            const vector & v1,
            cvector && v2 )  [friend]
```

7.8.4.26 operator- [6/8]

```
cvector operator- (
            cvector && v1,
            const cvector & v2 )  [friend]
```

7.8.4.27 operator- [7/8]

```
cvector operator- (
            cvector && v1,
            const vector & v2 )  [friend]
```

7.8.4.28 operator- [8/8]

```
cvector operator- (
            cvector && v1,
            cvector && v2 )  [friend]
```

7.8.4.29 operator/ [1/8]

```
cvector operator/ (
            complex a,
            const cvector & v )  [friend]
```

7.8.4.30 operator/ [2/8]

```
cvector operator/ (
            complex a,
            cvector && v )  [friend]
```

7.8.4.31 operator/ [3/8]

```
cvector operator/ (
            const cvector & v,
            complex a )  [friend]
```

7.8.4.32 operator/ [4/8]

```
cvector operator/ (
            const cvector & v,
            double x )  [friend]
```

7.8.4.33 operator/ [5/8]

```
cvector operator/ (
            cvector && v,
            complex a )  [friend]
```

7.8.4.34 operator/ [6/8]

```
cvector operator/ (
            cvector && v,
            double x )  [friend]
```

7.8.4.35 operator/ [7/8]

```
cvector operator/ (
            double x,
            const cvector & v )  [friend]
```

7.8.4.36 operator/ [8/8]

```
cvector operator/ (
            double x,
            cvector && v )  [friend]
```

7.8.4.37 operator== [1/3]

```
bool operator== (
            const cvector & v1,
            const cvector & v2 )  [friend]
```

7.8.4.38 operator== [2/3]

```
bool operator== (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.8.4.39 operator== [3/3]

```
bool operator== (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.8.5 Member Data Documentation

7.8.5.1 gvec

```
gsl_vector_complex* gsl::cvector::gvec  [protected]
```

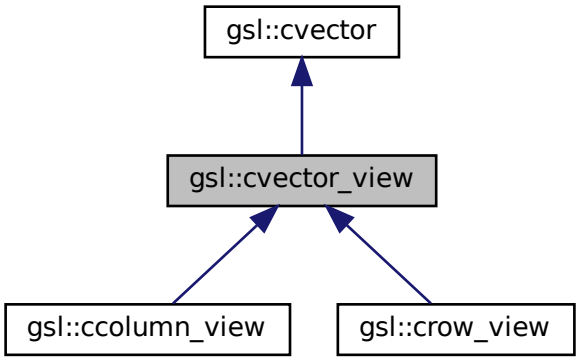
The documentation for this class was generated from the following files:

- /home/jspainhour/spheroidal_cpp/include/yawg/cvector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/cvector.cpp

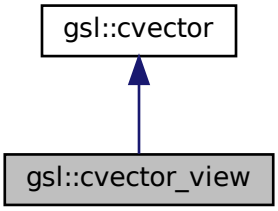
7.9 gsl::cvector_view Class Reference

`#include <cvector.h>`

Inheritance diagram for gsl::cvector_view:



Collaboration diagram for gsl::cvector_view:



Public Member Functions

- cvector_view (gsl_vector_complex ∗gvec_other)

  Constructor for vector_view pointing to data at gvec_other.
- ∼cvector_view ()
- cvector_view & operator= (const cvector &v)

  Assign data from cvector to a view.
- void clear ()

  Set all values in the view to zero.
- void resize (size_t n)

  Set all values in the view to zero.

Additional Inherited Members

7.9.1 Constructor & Destructor Documentation

7.9.1.1 cvector_view()

```
gsl::cvector_view::cvector_view (
            gsl_vector_complex * gvec_other )
```

Constructor for vector_view pointing to data at gvec_other.

Construct new gsl::cvector from gsl_vector_complex.

7.9.1.2 ~cvector_view()

```
gsl::cvector_view::~cvector_view ( )
```

7.9.2 Member Function Documentation

7.9.2.1 clear()

```
void gsl::cvector_view::clear ( )
```

Set all values in the view to zero.

7.9.2.2 operator=()

```
gsl::cvector_view & gsl::cvector_view::operator= (
            const cvector & v )
```

Assign data from cvector to a view.

7.9.2.3 resize()

```
void gsl::cvector_view::resize (
            size_t n )
```

Set all values in the view to zero.

The documentation for this class was generated from the following files:

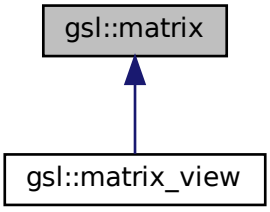
- /home/jspainhour/spheroidal_cpp/include/yawg/cvector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/cvector.cpp

7.10 gsl::matrix Class Reference

A wrapper class for gsl_matrix.

`#include <matrix.h>`

Inheritance diagram for gsl::matrix:



Public Member Functions

- matrix ()
  Construct empty matrix.
- matrix (size_t n, size_t m)
  Construct zero matrix of size n x m.
- matrix (FILE *in)
  Construct new gsl::matrix from .csv file.
- matrix (const vector &v)
  Construct new n x 1 gsl::matrix from gsl::vector.
- matrix (const matrix &M, size_t n, size_t m)
  Copy constructor creating n x m matrix.
- matrix (const matrix &M)
  Copy constructor.
- matrix (matrix &&M)
  Move constructor.
- matrix & operator= (const matrix &M)
- matrix & operator= (matrix &&M)
- matrix & operator+= (const matrix &M)
- matrix & operator-= (const matrix &M)
- matrix & operator*= (double x)
- matrix & operator/= (double x)
- matrix operator- () const
- ~matrix ()
- double & operator() (size_t i, size_t j)
- void set (size_t i, size_t j, double val)
- void set_col (size_t j, const vector &v)
- void set_row (size_t i, const vector &v)
- double operator() (size_t i, size_t j) const
- double get (size_t i, size_t j) const

- vector get_col (size_t j) const
- vector get_row (size_t i) const
- size_t size () const
- size_t nrows () const
- size_t ncols () const
- bool is_square () const
- gsl_matrix * get () const

  Access the pointer to the underlying gsl_matrix.
- void clear ()

  CLear the gsl::matrix, free underlying memory.
- void resize (size_t n, size_t m)

  Resize the gsl::matrix, freeing and allocating new memory.
- matrix reshape (size_t n, size_t m) const

  Return a new n x m gsl::matrix with same elements.
- matrix & T ()

  Replace the matrix with its transpose.
- void print (FILE *out=stdout) const

  Pretty-print the matrix to file stream.
- void print_csv (FILE *out=stdout) const

  Print the matrix to file stream in CSV format.
- void load_csv (FILE *in=stdin)

  Load the matrix from a file stream in CSV format.
- matrix_view view () const
- matrix_view submatrix (size_t i, size_t j, size_t n, size_t m) const
- row_view row (size_t i) const
- column_view column (size_t j) const

Protected Member Functions

- matrix (gsl_matrix *gmat)

  Construct new gsl::vector from gsl_vector's data.
- void gfree ()

  Private function to free allocated memory.
- void galloc (size_t n, size_t m)

  Private function to (continuously) allocate memory.

Protected Attributes

- gsl_matrix * gmat

Friends

- matrix operator* (double a, const matrix &M)
- matrix operator* (double a, matrix &&M)
- matrix operator* (const matrix &M, double a)
- matrix operator* (matrix &&M, double a)
- cmatrix operator* (complex a, const matrix &M)
- cmatrix operator* (const matrix &M, complex a)
- matrix operator/ (double a, const matrix &M)
- matrix operator/ (double a, matrix &&M)

- matrix operator/ (const matrix &M, double a)
- matrix operator/ (matrix &&M, double a)
- cmatrix operator/ (complex z, const matrix &M)
- cmatrix operator/ (const matrix &M, complex z)
- matrix operator+ (const matrix &M1, const matrix &M2)
- matrix operator+ (matrix &&M1, const matrix &M2)
- matrix operator+ (const matrix &M1, matrix &&M2)
- matrix operator+ (matrix &&M1, matrix &&M2)
- cmatrix operator+ (const matrix &M1, const cmatrix &M2)
- cmatrix operator+ (const matrix &M1, cmatrix &&M2)
- cmatrix operator+ (const cmatrix &M1, const matrix &M2)
- cmatrix operator+ (cmatrix &&M1, const matrix &M2)
- matrix operator- (const matrix &M1, const matrix &M2)
- matrix operator- (matrix &&M1, const matrix &M2)
- matrix operator- (const matrix &M1, matrix &&M2)
- matrix operator- (matrix &&M1, matrix &&M2)
- cmatrix operator- (const matrix &M1, const cmatrix &M2)
- cmatrix operator- (const matrix &M1, cmatrix &&M2)
- cmatrix operator- (const cmatrix &M1, const matrix &M2)
- cmatrix operator- (cmatrix &&M1, const matrix &M2)
- matrix operator∗ (const matrix &A, const matrix &B)
- bool operator== (const matrix &M1, const matrix &M2)
- bool operator!= (const matrix &M1, const matrix &M2)
- bool operator== (const matrix &M1, const cmatrix &M2)
- bool operator== (const cmatrix &M1, const matrix &M2)
- bool operator!= (const matrix &M1, const cmatrix &M2)
- bool operator!= (const cmatrix &M1, const matrix &M2)

7.10.1 Detailed Description

A wrapper class for gsl_matrix.

Stores and operates on a pointer to a gsl_matrix.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 matrix() [1/8]

```
gsl::matrix::matrix ( )
```

Construct empty matrix.

7.10.2.2 matrix() [2/8]

```
gsl::matrix::matrix (
            size_t n,
            size_t m )
```

Construct zero matrix of size n x m.

Parameters

n	Number of rows
m	Number of columns

Note

By convention, all "empty" matrices have nullprt data

7.10.2.3 matrix() [3/8]

```
gsl::matrix::matrix (
            FILE * in )
```

Construct new gsl::matrix from .csv file.

Parameters

in	stdio.h file handle

7.10.2.4 matrix() [4/8]

```
gsl::matrix::matrix (
            const vector & v )
```

Construct new n x 1 gsl::matrix from gsl::vector.

Parameters

v	Vector to copy

7.10.2.5 matrix() [5/8]

```
gsl::matrix::matrix (
            const matrix & M,
            size_t n,
            size_t m )
```

Copy constructor creating n x m matrix.

Parameters

M	gsl::matrix to copy
n	Number of rows
m	Number of columns

7.10.2.6 matrix() [6/8]

```
gsl::matrix::matrix (
            const matrix & M )
```

Copy constructor.

Parameters

M	gsl::matrix to copy

7.10.2.7 matrix() [7/8]

```
gsl::matrix::matrix (
            gsl::matrix && M )
```

Move constructor.

Parameters

M	gsl::matrix to move

7.10.2.8 ~matrix()

```
gsl::matrix::~matrix ( )
```

7.10.2.9 matrix() [8/8]

```
gsl::matrix::matrix (
            gsl_matrix * gmat )  [protected]
```

Construct new gsl::vector from gsl_vector's data.

7.10.3 Member Function Documentation

7.10.3.1 clear()

```
void gsl::matrix::clear ( )
```

CLear the gsl::matrix, free underlying memory.

7.10.3.2 column()

```
gsl::column_view gsl::matrix::column (
            size_t j ) const
```

7.10.3.3 galloc()

```
void gsl::matrix::galloc (
            size_t n,
            size_t m )  [protected]
```

Private function to (continuously) allocate memory.

Note

This method allocates contiguous, zero-initialized memory. This is slightly slower than using gsl_matrix_alloc, but allows for intuitive usage of row views.

7.10.3.4 get() [1/2]

```
gsl_matrix* gsl::matrix::get ( ) const  [inline]
```

Access the pointer to the underlying gsl_matrix.

7.10.3.5 get() [2/2]

```
double gsl::matrix::get (
            size_t i,
            size_t j ) const
```

7.10.3.6 get_col()

```
gsl::vector gsl::matrix::get_col (
            size_t j ) const
```

7.10.3.7 get_row()

```
gsl::vector gsl::matrix::get_row (
            size_t i ) const
```

7.10.3.8 gfree()

```
void gsl::matrix::gfree ( )  [protected]
```

Private function to free allocated memory.

7.10.3.9 is_square()

```
bool gsl::matrix::is_square ( ) const
```

7.10.3.10 load_csv()

```
void gsl::matrix::load_csv (
            FILE * in = stdin )
```

Load the matrix from a file stream in CSV format.

Parameters

in	File stream to load from

7.10.3.11 ncols()

```
size_t gsl::matrix::ncols ( ) const
```

7.10.3.12 nrows()

```
size_t gsl::matrix::nrows ( ) const
```

7.10.3.13 operator()() [1/2]

```
double & gsl::matrix::operator() (
            size_t i,
            size_t j )
```

7.10.3.14 operator()() [2/2]

```
double gsl::matrix::operator() (
            size_t i,
            size_t j ) const
```

7.10.3.15 operator*=()

```
gsl::matrix & gsl::matrix::operator*= (
            double x )
```

7.10.3.16 operator+=()

```
gsl::matrix & gsl::matrix::operator+= (
            const matrix & M )
```

7.10.3.17 operator-()

```
gsl::matrix gsl::matrix::operator- ( ) const
```

7.10.3.18 operator-=()

```
gsl::matrix & gsl::matrix::operator-= (
            const matrix & M )
```

7.10.3.19 operator/=()

```
gsl::matrix & gsl::matrix::operator/= (
            double x )
```

7.10.3.20 operator=() [1/2]

```
gsl::matrix & gsl::matrix::operator= (
            const matrix & M )
```

7.10.3.21 operator=() [2/2]

```
gsl::matrix & gsl::matrix::operator= (
            gsl::matrix && M )
```

7.10.3.22 print()

```
void gsl::matrix::print (
            FILE * out = stdout ) const
```

Pretty-print the matrix to file stream.

Parameters

out	File stream to print to

7.10.3.23 print_csv()

```
void gsl::matrix::print_csv (
            FILE * out = stdout ) const
```

Print the matrix to file stream in CSV format.

Parameters

out	File stream to print to

7.10.3.24 reshape()

```
gsl::matrix gsl::matrix::reshape (
            size_t n,
            size_t m ) const
```

Return a new n x m gsl::matrix with same elements.

Parameters

n	Number of rows
m	Number of columns

Returns

New gsl::matrix with same elements

7.10.3.25 resize()

```
void gsl::matrix::resize (
            size_t n,
            size_t m )
```

Resize the gsl::matrix, freeing and allocating new memory.

Resize the gsl::matrix, setting elements to zero.

Parameters

n	Number of rows
m	Number of columns

Note

This function will always free and reallocate memory, setting the elements to zero.

7.10.3.26 row()

```
gsl::row_view gsl::matrix::row (
            size_t i ) const
```

7.10.3.27 set()

```
void gsl::matrix::set (
            size_t i,
            size_t j,
            double val )
```

7.10.3.28 set_col()

```
void gsl::matrix::set_col (
            size_t j,
            const vector & v )
```

7.10.3.29 set_row()

```
void gsl::matrix::set_row (
            size_t i,
            const vector & v )
```

7.10.3.30 size()

```
size_t gsl::matrix::size ( ) const
```

7.10.3.31 submatrix()

```
gsl::matrix_view gsl::matrix::submatrix (
            size_t i,
            size_t j,
            size_t n,
            size_t m ) const
```

7.10.3.32 T()

```
gsl::matrix & gsl::matrix::T ( )
```

Replace the matrix with its transpose.

Compute the matrix transpose, in-place if square.

7.10.3.33 view()

```
gsl::matrix_view gsl::matrix::view ( ) const
```

7.10.4 Friends And Related Function Documentation

7.10.4.1 operator"!= [1/3]

```
bool operator!= (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.10.4.2 operator"!= [2/3]

```
bool operator!= (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.10.4.3 operator"!= [3/3]

```
bool operator!= (
            const matrix & M1,
            const matrix & M2 )  [friend]
```

7.10.4.4 operator* [1/7]

```
cmatrix operator* (
            complex a,
            const matrix & M )  [friend]
```

7.10.4.5 operator* [2/7]

```
matrix operator* (
            const matrix & A,
            const matrix & B )  [friend]
```

7.10.4.6 operator* [3/7]

```
cmatrix operator* (
            const matrix & M,
            complex a )  [friend]
```

7.10.4.7 operator* [4/7]

```
matrix operator* (
            const matrix & M,
            double a )  [friend]
```

7.10.4.8 operator* [5/7]

```
matrix operator* (
            double a,
            const matrix & M )  [friend]
```

7.10.4.9 operator* [6/7]

```
matrix operator* (
            double a,
            matrix && M )  [friend]
```

7.10.4.10 operator* [7/7]

```
matrix operator* (
            matrix && M,
            double a )  [friend]
```

7.10.4.11 operator+ [1/8]

```
cmatrix operator+ (
            cmatrix && M1,
            const matrix & M2 )  [friend]
```

7.10.4.12 operator+ [2/8]

```
cmatrix operator+ (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.10.4.13 operator+ [3/8]

```
cmatrix operator+ (
            const matrix & M1,
            cmatrix && M2 )  [friend]
```

7.10.4.14 operator+ [4/8]

```
cmatrix operator+ (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.10.4.15 operator+ [5/8]

```
matrix operator+ (
            const matrix & M1,
            const matrix & M2 )  [friend]
```

7.10.4.16 operator+ [6/8]

```
matrix operator+ (
            const matrix & M1,
            matrix && M2 )  [friend]
```

7.10.4.17 operator+ [7/8]

```
matrix operator+ (
            matrix && M1,
            const matrix & M2 )  [friend]
```

7.10.4.18 operator+ [8/8]

```
matrix operator+ (
            matrix && M1,
            matrix && M2 )  [friend]
```

7.10.4.19 operator- [1/8]

```
cmatrix operator- (
            cmatrix && M1,
            const matrix & M2 )  [friend]
```

7.10.4.20 operator- [2/8]

```
cmatrix operator- (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.10.4.21 operator- [3/8]

```
cmatrix operator- (
            const matrix & M1,
            cmatrix && M2 )  [friend]
```

7.10.4.22 operator- [4/8]

```
cmatrix operator- (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.10.4.23 operator- [5/8]

```
matrix operator- (
            const matrix & M1,
            const matrix & M2 )  [friend]
```

7.10.4.24 operator- [6/8]

```
matrix operator- (
            const matrix & M1,
            matrix && M2 )  [friend]
```

7.10.4.25 operator- [7/8]

```
matrix operator- (
            matrix && M1,
            const matrix & M2 )  [friend]
```

7.10.4.26 operator- [8/8]

```
matrix operator- (
            matrix && M1,
            matrix && M2 )  [friend]
```

7.10.4.27 operator/ [1/6]

```
cmatrix operator/ (
            complex z,
            const matrix & M )  [friend]
```

7.10.4.28 operator/ [2/6]

```
cmatrix operator/ (
            const matrix & M,
            complex z )  [friend]
```

7.10.4.29 operator/ [3/6]

```
matrix operator/ (
            const matrix & M,
            double a )  [friend]
```

7.10.4.30 operator/ [4/6]

```
matrix operator/ (
            double a,
            const matrix & M )  [friend]
```

7.10.4.31 operator/ [5/6]

```
matrix operator/ (
            double a,
            matrix && M )  [friend]
```

7.10.4.32 operator/ [6/6]

```
matrix operator/ (
            matrix && M,
            double a )  [friend]
```

7.10.4.33 operator== [1/3]

```
bool operator== (
            const cmatrix & M1,
            const matrix & M2 )  [friend]
```

7.10.4.34 operator== [2/3]

```
bool operator== (
            const matrix & M1,
            const cmatrix & M2 )  [friend]
```

7.10.4.35 operator== [3/3]

```
bool operator== (
            const matrix & M1,
            const matrix & M2 )  [friend]
```

7.10.5 Member Data Documentation

7.10.5.1 gmat

```
gsl_matrix* gsl::matrix::gmat  [protected]
```

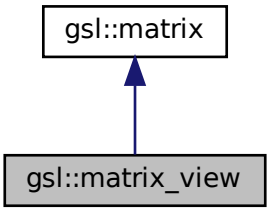
The documentation for this class was generated from the following files:

- /home/jspainhour/spheroidal_cpp/include/yawg/matrix.h
- /home/jspainhour/spheroidal_cpp/src_yawg/matrix.cpp

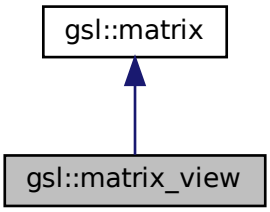
7.11 gsl::matrix_view Class Reference

```
#include <matrix.h>
```

Inheritance diagram for gsl::matrix_view:



Collaboration diagram for gsl::matrix_view:



Public Member Functions

- matrix_view (gsl_matrix ∗gvec_other)

  Constructor for vector_view pointing to data at gvec_other.
- ∼matrix_view ()
- matrix_view & operator= (const matrix &M)

  Assign data from matrix to view.
- void clear ()
- void resize (size_t n, size_t m)

Additional Inherited Members

7.11.1 Constructor & Destructor Documentation

7.11.1.1 matrix_view()

```
gsl::matrix_view::matrix_view (
            gsl_matrix * gvec_other )
```

Constructor for vector_view pointing to data at gvec_other.

7.11.1.2 ∼matrix_view()

```
gsl::matrix_view::~matrix_view ( )
```

7.11.2 Member Function Documentation

7.11.2.1 clear()

```
void gsl::matrix_view::clear ( )
```

7.11.2.2 operator=()

```
gsl::matrix_view & gsl::matrix_view::operator= (
            const matrix & M )
```

Assign data from matrix to view.

7.11.2.3 resize()

```
void gsl::matrix_view::resize (
            size_t n,
            size_t m )
```

The documentation for this class was generated from the following files:

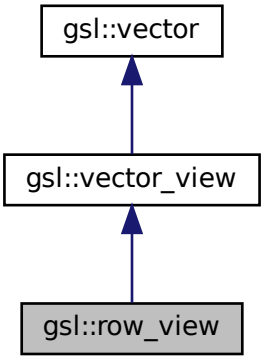
- /home/jspainhour/spheroidal_cpp/include/yawg/matrix.h
- /home/jspainhour/spheroidal_cpp/src_yawg/matrix.cpp

7.12 gsl::row_view Class Reference

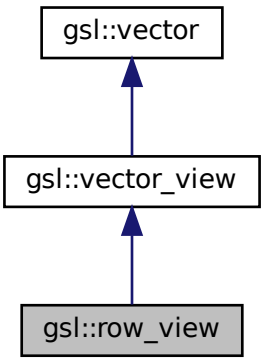
A subclass of cvector_view for stride-1 cvectors.

`#include <vector.h>`

Inheritance diagram for gsl::row_view:



Collaboration diagram for gsl::row_view:



Public Member Functions

- row_view (gsl_vector *gvec_other)

  Construct row_view from existing vector view, checking that stride is 1.
- row_view & operator= (const vector &v)

  Assign data from vector to view.
- matrix_view reshape (size_t n, size_t m) const

  Return a matrix view out of the elements of the row.

Additional Inherited Members

7.12.1 Detailed Description

A subclass of cvector_view for stride-1 cvectors.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 row_view()

```
gsl::row_view::row_view (
            gsl_vector * gvec_other )  [inline]
```

Construct row_view from existing vector view, checking that stride is 1.

7.12.3 Member Function Documentation

7.12.3.1 operator=()

```
gsl::row_view & gsl::row_view::operator= (
            const vector & v )
```

Assign data from vector to view.

7.12.3.2 reshape()

```
gsl::matrix_view gsl::row_view::reshape (
            size_t n,
            size_t m ) const
```

Return a matrix view out of the elements of the row.

The documentation for this class was generated from the following files:

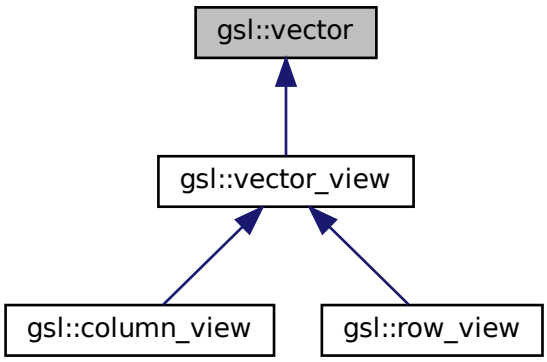
- /home/jspainhour/spheroidal_cpp/include/yawg/vector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/vector.cpp

7.13 gsl::vector Class Reference

A wrapper class for gsl_vector.

`#include <vector.h>`

Inheritance diagram for gsl::vector:



Public Member Functions

- vector ()
  Construct empty vector.
- vector (size_t n)
  Construct zero vector of size n.
- vector (const vector &v)
- vector (vector &&v)
- vector & operator= (const vector &v)
- vector & operator= (vector &&v)
- vector & operator+= (const vector &v)
- vector & operator-= (const vector &v)
- vector & operator∗= (double a)
- vector & operator/= (double a)
- vector operator- () const
- ∼vector ()
- double & operator() (size_t i)
- void set (size_t i, double val)
- double operator() (size_t i) const
- double get (size_t i) const
- size_t size () const
- gsl_vector ∗ get () const
  Access the pointer to the underlying gsl_vector.
- void resize (size_t n)
  Resize the gsl::vector, setting elements to zero.
- void clear ()

Clear the gsl::vector, free underlying memory.

- void print (FILE *out=stdout) const

  Pretty-print the vector to file stream.

- double norm () const

  Return the 2-norm of the vector.

- vector_view view () const
- vector_view subvector (size_t offset, size_t n) const

Protected Member Functions

- vector (gsl_vector *gvec_other)

  Construct new gsl::vector from gsl_vector.

- void gfree ()

  Private function to free allocated memory.

- void galloc (size_t n)

  Private function to (continuously) allocate memory.

Protected Attributes

- gsl_vector * gvec

Friends

- vector operator* (double a, const vector &v)
- vector operator* (double a, vector &&v)
- vector operator* (const vector &v, double a)
- vector operator* (vector &&v, double a)
- cvector operator* (complex a, const vector &v)
- cvector operator* (const vector &v, complex a)
- vector operator/ (double a, const vector &v)
- vector operator/ (double a, vector &&v)
- vector operator/ (const vector &v, double a)
- vector operator/ (vector &&v, double a)
- cvector operator/ (complex z, const vector &v)
- cvector operator/ (const vector &v, complex z)
- vector operator+ (const vector &v1, const vector &v2)
- vector operator+ (vector &&v1, const vector &v2)
- vector operator+ (const vector &v1, vector &&v2)
- vector operator+ (vector &&v1, vector &&v2)
- cvector operator+ (const vector &v1, const cvector &v2)
- cvector operator+ (const vector &v1, cvector &&v2)
- cvector operator+ (const cvector &v1, const vector &v2)
- cvector operator+ (cvector &&v1, const vector &v2)
- vector operator- (const vector &v1, const vector &v2)
- vector operator- (vector &&v1, const vector &v2)
- vector operator- (const vector &v1, vector &&v2)
- vector operator- (vector &&v1, vector &&v2)
- cvector operator- (const vector &v1, const cvector &v2)
- cvector operator- (const vector &v1, cvector &&v2)
- cvector operator- (const cvector &v1, const vector &v2)
- cvector operator- (cvector &&v1, const vector &v2)

- bool operator== (const vector &v1, const vector &v2)
- bool operator!= (const vector &v1, const vector &v2)
- bool operator== (const vector &v1, const cvector &v2)
- bool operator== (const cvector &v1, const vector &v2)
- bool operator!= (const cvector &v1, const vector &v2)
- bool operator!= (const vector &v1, const cvector &v2)
- vector operator* (const matrix &M, const vector &v)

7.13.1 Detailed Description

A wrapper class for gsl_vector.

Stores and operates on a pointer to a gsl_vector.

7.13.2 Constructor & Destructor Documentation

7.13.2.1 vector() [1/5]

```
gsl::vector::vector ( )
```

Construct empty vector.

7.13.2.2 vector() [2/5]

```
gsl::vector::vector (
            size_t n )  [explicit]
```

Construct zero vector of size n.

7.13.2.3 vector() [3/5]

```
gsl::vector::vector (
            const vector & v )
```

7.13.2.4 vector() [4/5]

```
gsl::vector::vector (
            gsl::vector && v )
```

7.13.2.5 ~vector()

```
gsl::vector::~vector ( )
```

7.13.2.6 vector() [5/5]

```
gsl::vector::vector (
            gsl_vector * gvec_other )  [protected]
```

Construct new gsl::vector from gsl_vector.

Construct new gsl::vector from gsl_vector's data.

7.13.3 Member Function Documentation

7.13.3.1 clear()

```
void gsl::vector::clear ( )
```

Clear the gsl::vector, free underlying memory.

7.13.3.2 galloc()

```
void gsl::vector::galloc (
            size_t n )  [protected]
```

Private function to (continuously) allocate memory.

Note

This method allocates contiguous, zero-initialized memory. This is slightly slower than using gsl_vector_alloc, but allows for intuitive usage of row views.

7.13.3.3 get() [1/2]

```
gsl_vector* gsl::vector::get ( ) const  [inline]
```

Access the pointer to the underlying gsl_vector.

7.13.3.4 get() [2/2]

```
double gsl::vector::get (
            size_t i ) const
```

7.13.3.5 gfree()

```
void gsl::vector::gfree ( )  [protected]
```

Private function to free allocated memory.

7.13.3.6 norm()

```
double gsl::vector::norm ( ) const  [inline]
```

Return the 2-norm of the vector.

7.13.3.7 operator()() [1/2]

```
double & gsl::vector::operator() (
            size_t i )
```

7.13.3.8 operator()() [2/2]

```
double gsl::vector::operator() (
            size_t i ) const
```

7.13.3.9 operator*=()

```
gsl::vector & gsl::vector::operator*= (
            double a )
```

7.13.3.10 operator+=()

```
gsl::vector & gsl::vector::operator+= (
            const vector & v )
```

7.13.3.11 operator-()

```
gsl::vector gsl::vector::operator- ( ) const
```

7.13.3.12 operator-=()

```
gsl::vector & gsl::vector::operator-= (
            const vector & v )
```

7.13.3.13 operator/=()

```
gsl::vector & gsl::vector::operator/= (
            double a )
```

7.13.3.14 operator=() [1/2]

```
gsl::vector & gsl::vector::operator= (
            const vector & v )
```

7.13.3.15 operator=() [2/2]

```
gsl::vector & gsl::vector::operator= (
            gsl::vector && v )
```

7.13.3.16 print()

```
void gsl::vector::print (
            FILE * out = stdout ) const
```

Pretty-print the vector to file stream.

Parameters

out	File stream to print to

7.13.3.17 resize()

```
void gsl::vector::resize (
            size_t n )
```

Resize the gsl::vector, setting elements to zero.

Parameters

n	Number of elements

Note

This function will always free and reallocate memory, setting the elements to zero.

7.13.3.18 set()

```
void gsl::vector::set (
            size_t i,
            double val )
```

7.13.3.19 size()

```
size_t gsl::vector::size ( ) const
```

7.13.3.20 subvector()

```
gsl::vector_view gsl::vector::subvector (
            size_t offset,
            size_t n ) const
```

7.13.3.21 view()

```
gsl::vector_view gsl::vector::view ( ) const
```

7.13.4 Friends And Related Function Documentation

7.13.4.1 operator"!= [1/3]

```
bool operator!= (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.13.4.2 operator"!= [2/3]

```
bool operator!= (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.13.4.3 operator"!= [3/3]

```
bool operator!= (
            const vector & v1,
            const vector & v2 )  [friend]
```

7.13.4.4 operator* [1/7]

```
cvector operator* (
            complex a,
            const vector & v )  [friend]
```

7.13.4.5 operator* [2/7]

```
vector operator* (
            const matrix & M,
            const vector & v )  [friend]
```

7.13.4.6 operator* [3/7]

```
cvector operator* (
            const vector & v,
            complex a )  [friend]
```

7.13.4.7 operator* [4/7]

```
vector operator* (
            const vector & v,
            double a )  [friend]
```

7.13.4.8 operator* [5/7]

```
vector operator* (
            double a,
            const vector & v )  [friend]
```

7.13.4.9 operator* [6/7]

```
vector operator* (
            double a,
            vector && v )  [friend]
```

7.13.4.10 operator* [7/7]

```
vector operator* (
            vector && v,
            double a )  [friend]
```

7.13.4.11 operator+ [1/8]

```
cvector operator+ (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.13.4.12 operator+ [2/8]

```
cvector operator+ (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.13.4.13 operator+ [3/8]

```
vector operator+ (
            const vector & v1,
            const vector & v2 )  [friend]
```

7.13.4.14 operator+ [4/8]

```
cvector operator+ (
            const vector & v1,
            cvector && v2 )  [friend]
```

7.13.4.15 operator+ [5/8]

```
vector operator+ (
            const vector & v1,
            vector && v2 )  [friend]
```

7.13.4.16 operator+ [6/8]

```
cvector operator+ (
            cvector && v1,
            const vector & v2 )  [friend]
```

7.13.4.17 operator+ [7/8]

```
vector operator+ (
            vector && v1,
            const vector & v2 )  [friend]
```

7.13.4.18 operator+ [8/8]

```
vector operator+ (
            vector && v1,
            vector && v2 )  [friend]
```

7.13.4.19 operator- [1/8]

```
cvector operator- (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.13.4.20 operator- [2/8]

```
cvector operator- (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.13.4.21 operator- [3/8]

```
vector operator- (
            const vector & v1,
            const vector & v2 )  [friend]
```

7.13.4.22 operator- [4/8]

```
cvector operator- (
            const vector & v1,
            cvector && v2 )  [friend]
```

7.13.4.23 operator- [5/8]

```
vector operator- (
            const vector & v1,
            vector && v2 )  [friend]
```

7.13.4.24 operator- [6/8]

```
cvector operator- (
            cvector && v1,
            const vector & v2 )  [friend]
```

7.13.4.25 operator- [7/8]

```
vector operator- (
            vector && v1,
            const vector & v2 )  [friend]
```

7.13.4.26 operator- [8/8]

```
vector operator- (
            vector && v1,
            vector && v2 )  [friend]
```

7.13.4.27 operator/ [1/6]

```
cvector operator/ (
            complex z,
            const vector & v )  [friend]
```

7.13.4.28 operator/ [2/6]

```
cvector operator/ (
            const vector & v,
            complex z )  [friend]
```

7.13.4.29 operator/ [3/6]

```
vector operator/ (
            const vector & v,
            double a )  [friend]
```

7.13.4.30 operator/ [4/6]

```
vector operator/ (
            double a,
            const vector & v )  [friend]
```

7.13.4.31 operator/ [5/6]

```
vector operator/ (
            double a,
            vector && v )  [friend]
```

7.13.4.32 operator/ [6/6]

```
vector operator/ (
            vector && v,
            double a )  [friend]
```

7.13.4.33 operator== [1/3]

```
bool operator== (
            const cvector & v1,
            const vector & v2 )  [friend]
```

7.13.4.34 operator== [2/3]

```
bool operator== (
            const vector & v1,
            const cvector & v2 )  [friend]
```

7.13.4.35 operator== [3/3]

```
bool operator== (
            const vector & v1,
            const vector & v2 )  [friend]
```

7.13.5 Member Data Documentation

7.13.5.1 gvec

```
gsl_vector* gsl::vector::gvec  [protected]
```

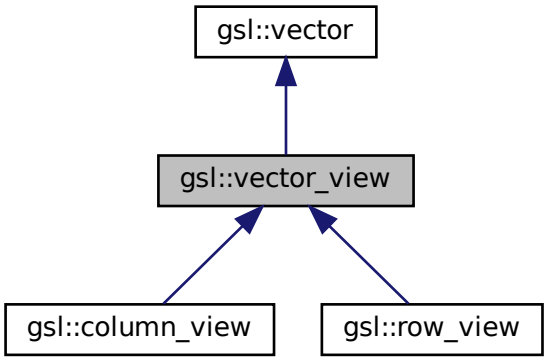
The documentation for this class was generated from the following files:

- /home/jspainhour/spheroidal_cpp/include/yawg/vector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/vector.cpp

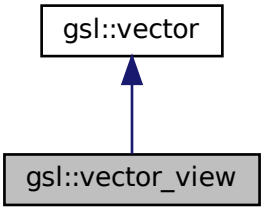
7.14 gsl::vector_view Class Reference

`#include <vector.h>`

Inheritance diagram for gsl::vector_view:



Collaboration diagram for gsl::vector_view:



Public Member Functions

- vector_view (gsl_vector *gvec_other)

  Constructor for vector_view pointing to data at gvec_other.
- ~vector_view ()
- vector_view & operator= (const vector &v)

  Assign data from vector to view.
- void clear ()

  Set all values in the view to zero.
- void resize (size_t n)

  Set all values in the view to zero.

Additional Inherited Members

7.14.1 Constructor & Destructor Documentation

7.14.1.1 vector_view()

```
gsl::vector_view::vector_view (
            gsl_vector * gvec_other )
```

Constructor for vector_view pointing to data at gvec_other.

Construct new gsl::vector from gsl_vector.

7.14.1.2 ~vector_view()

```
gsl::vector_view::~vector_view ( )
```

7.14.2 Member Function Documentation

7.14.2.1 clear()

```
void gsl::vector_view::clear ( )
```

Set all values in the view to zero.

7.14.2.2 operator=()

```
gsl::vector_view & gsl::vector_view::operator= (
            const vector & v )
```

Assign data from vector to view.

7.14.2.3 resize()

```
void gsl::vector_view::resize (
            size_t n )
```

Set all values in the view to zero.

The documentation for this class was generated from the following files:

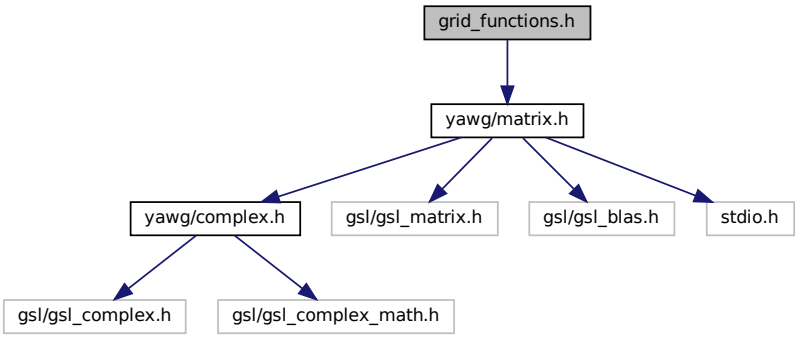
- /home/jspainhour/spheroidal_cpp/include/yawg/vector.h
- /home/jspainhour/spheroidal_cpp/src_yawg/vector.cpp

Chapter 8

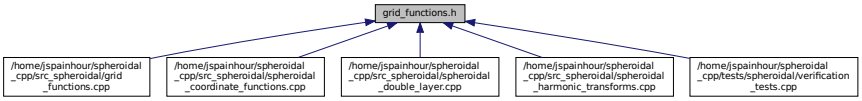
File Documentation

8.1 grid_functions.h File Reference

`#include <yawg/matrix.h>`
Include dependency graph for grid_functions.h:



This graph shows which files directly or indirectly include this file:



Functions

- int spharm_grid_size_ord (int p, int &nu, int &nv)

  Computes grid size of spheroidal harmonics grid given the order.
- int spharm_grid_size_tot (int ntot, int &nu, int &nv)

  Computes grid size of spheroidal harmonics grid given the total number of points.
- void gl_grid (size_t p, gsl::matrix &U, gsl::matrix &V)

  Populates the matrices `U` and `V` with a spheroidal harmonics grid of order `p`.

8.1.1 Function Documentation

8.1.1.1 gl_grid()

```
void gl_grid (
            size_t p,
            gsl::matrix & U,
            gsl::matrix & V )
```

Populates the matrices U and V with a spheroidal harmonics grid of order p.

Populates the matrices U and V with a spheroidal harmonics grid of order p.

Parameters

p	Order of the spheroidal harmonics
U	Reference to gsl::matrix of Gaussian spaced rows
V	Reference to gsl::matrix of Uniform spaced columns

Mapped to a spheroid, u is spaced on [0, pi] and v is spaced on [0, 2pi].

8.1.1.2 spharm_grid_size_ord()

```
int spharm_grid_size_ord (
            int p,
            int & nu,
            int & nv )
```

Computes grid size of spheroidal harmonics grid given the order.

Parameters

p	Order of the spheroidal harmonics
nu	Number of points in the theta direction
nv	Number of points in the lambda direction

Returns

The order of the spheroidal harmonics (legacy usage)

8.1.1.3 spharm_grid_size_tot()

```
int spharm_grid_size_tot (
            int ntot,
```

```
int & nu,
int & nv )
```

Computes grid size of spheroidal harmonics grid given the total number of points.

Parameters

ntot	The total number of points in the grid
nu	Number of points in the theta direction
nv	Number of points in the lambda direction

Note

Will cause an error if the number of points is not valid for a spheroidal harmonics grid

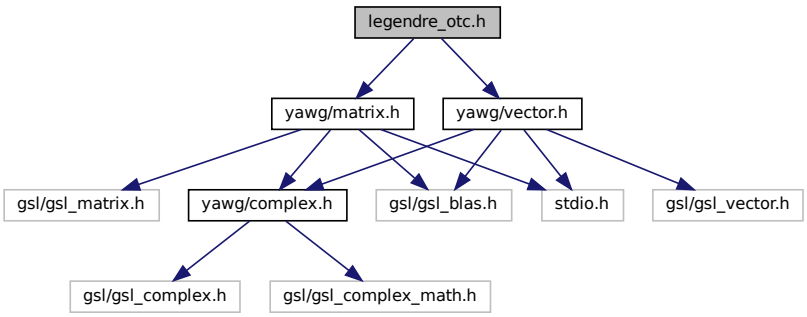
Returns

The order of the spheroidal harmonics (legacy usage)

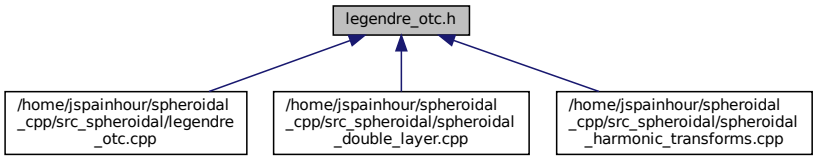
8.2 legendre_otc.h File Reference

```
#include <yawg/vector.h>
#include <yawg/matrix.h>
```

Include dependency graph for legendre_otc.h:



This graph shows which files directly or indirectly include this file:



Functions

- gsl::vector cont_frac (int n, int m, gsl::vector u)
- int geti (int n, int m)
- void legendre_otc (int p, gsl::vector u, gsl::matrix &P)

  Computes associated Legendre functions off the cut of the first kind, $$P_n^m(u)$$, $$u>1$$.
- void legendre_otc (int p, gsl::vector u, gsl::matrix &P, gsl::matrix &Q)

  Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, for $$u>1$$.
- void Dlegendre_otc (int p, gsl::vector u, gsl::matrix &P, gsl::matrix &dP)

  Computes associated legendre functions off the cut of the first kind, $$P_n^m(u)$$ and their derivatives.
- void Dlegendre_otc (int p, gsl::vector u, gsl::matrix &P, gsl::matrix &Q, gsl::matrix &dP, gsl::matrix &dQ)

  Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, and their derivatives, for $$u>1$$.

8.2.1 Function Documentation

8.2.1.1 cont_frac()

```
gsl::vector cont_frac (
            int n,
            int m,
            gsl::vector u )
```

8.2.1.2 Dlegendre_otc() [1/2]

```
void Dlegendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P,
            gsl::matrix & dP )
```

Computes associated legendre functions off the cut of the first kind, $$P_n^m(u)$$ and their derivatives.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions
P	Matrix of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
dP	Matrix of derivatives of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

Note

Will cause an error if any u-value is less than or equal to 1.

8.2.1.3 Dlegendre_otc() [2/2]

```
void Dlegendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P,
            gsl::matrix & Q,
            gsl::matrix & dP,
            gsl::matrix & dQ )
```

Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, and their derivatives, for $$u>1$$.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions
P	Matrix of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
Q	Matrix of associated legendre functions off the cut of the second kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
dP	Matrix of derivatives of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
dQ	Matrix of derivatives of associated legendre functions off the cut of the second kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

Note

Will cause an error if any u-value is less than or equal to 1.

8.2.1.4 geti()

```
int geti (
            int n,
            int m )
```

8.2.1.5 legendre_otc() [1/2]

```
void legendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P )
```

Computes associated Legendre functions off the cut of the first kind, $$P_n^m(u)$$, $$u>1$$.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions
P	Matrix to store associated Legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

Note

Will cause an error if any u-value is less than or equal to 1.

8.2.1.6 legendre_otc() [2/2]

```
void legendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P,
            gsl::matrix & Q )
```

Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, for $$u>1$$.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions
P	Matrix of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
Q	Matrix of associated legendre functions off the cut of the second kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

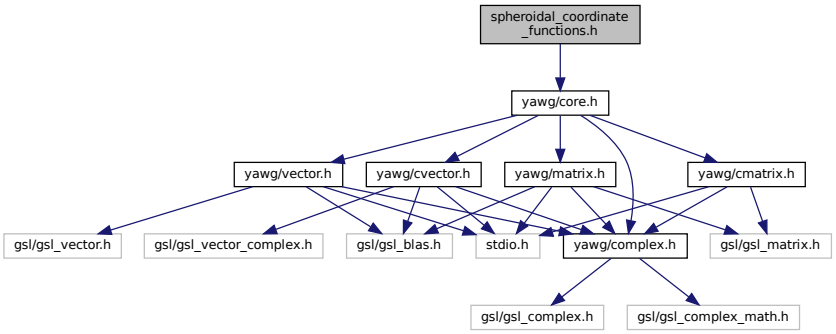
Note

Will cause an error if any u-value is less than or equal to 1.

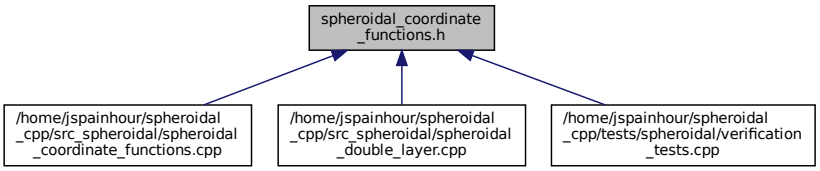
8.3 spheroidal_coordinate_functions.h File Reference

```
#include <yawg/core.h>
```

Include dependency graph for spheroidal_coordinate_functions.h:



This graph shows which files directly or indirectly include this file:



Functions

- gsl::matrix spheroidal_to_cart (gsl::matrix S, double a)
- gsl::matrix cart_to_spheroidal (gsl::matrix X, double a)

8.3.1 Function Documentation

8.3.1.1 cart_to_spheroidal()

```
gsl::matrix cart_to_spheroidal (
            gsl::matrix X,
            double a )
```

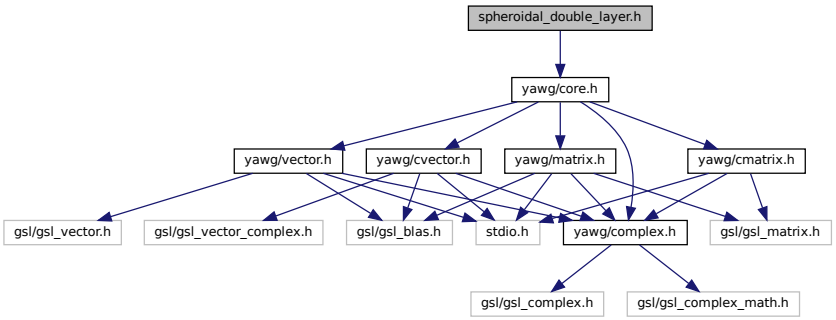
8.3.1.2 spheroidal_to_cart()

```
gsl::matrix spheroidal_to_cart (
            gsl::matrix S,
            double a )
```

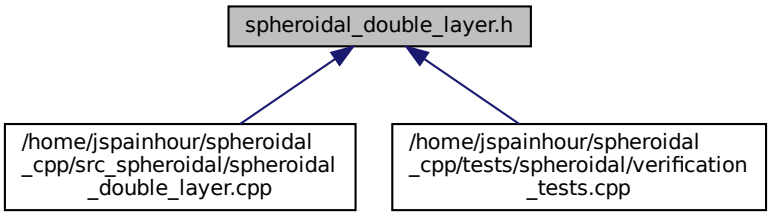
8.4 spheroidal_double_layer.h File Reference

```
#include <yawg/core.h>
```

Include dependency graph for spheroidal_double_layer.h:



This graph shows which files directly or indirectly include this file:



Functions

- gsl::matrix solid_harmonic (int p, gsl::vector u_x, int region=0)
- void DLspectrum (int p, double u0, gsl::vector &lambda_int, gsl::vector &lambda_surf, gsl::vector &lambda_ext)
- gsl::cmatrix Ynm_matrix (int p, gsl::vector v, gsl::vector phi)
- gsl::cmatrix spheroidal_double_layer (gsl::cmatrix sigma, double u0, gsl::matrix X, int target_coords=0)
- gsl::cmatrix spheroidal_double_layer (gsl::cmatrix sigma, double u0)

8.4.1 Function Documentation

8.4.1.1 DLspectrum()

```
void DLspectrum (
            int p,
            double u0,
            gsl::vector & lambda_int,
            gsl::vector & lambda_surf,
            gsl::vector & lambda_ext )
```

8.4.1.2 solid_harmonic()

```
gsl::matrix solid_harmonic (
            int p,
            gsl::vector u_x,
            int region = 0 )
```

8.4.1.3 spheroidal_double_layer() [1/2]

```
gsl::cmatrix spheroidal_double_layer (
            gsl::cmatrix sigma,
            double u0 )
```

8.4.1.4 spheroidal_double_layer() [2/2]

```
gsl::cmatrix spheroidal_double_layer (
            gsl::cmatrix sigma,
            double u0,
            gsl::matrix X,
            int target_coords = 0 )
```

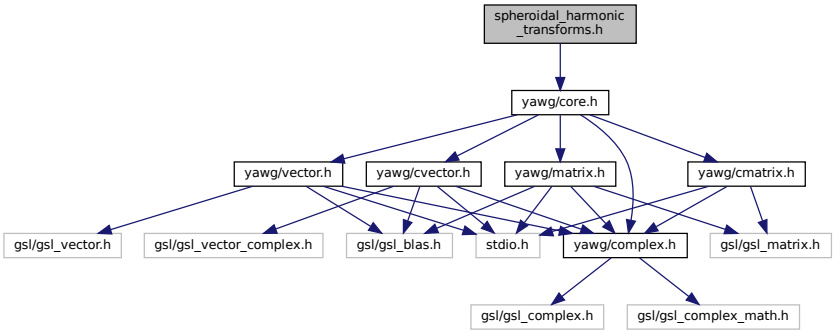
8.4.1.5 Ynm_matrix()

```
gsl::cmatrix Ynm_matrix (
            int p,
            gsl::vector v,
            gsl::vector phi )
```

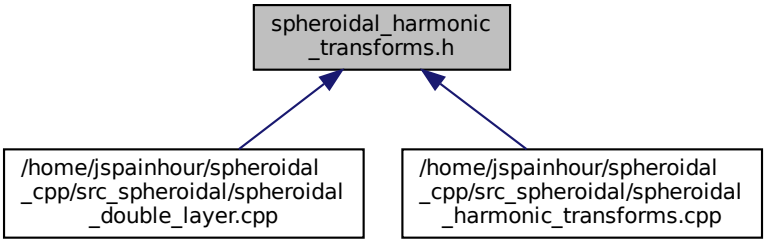
8.5 spheroidal_harmonic_transforms.h File Reference

```
#include <yawg/core.h>
```

Include dependency graph for spheroidal_harmonic_transforms.h:



This graph shows which files directly or indirectly include this file:



Functions

- gsl::matrix get_legendre_matrix (int p, int m)
- gsl::matrix get_legendre_matrix_inv (int p, int m)
- gsl::cmatrix spheroidal_analysis (gsl::cmatrix f)
- gsl::cmatrix spheroidal_snythesis (gsl::cmatrix shc)

8.5.1 Function Documentation

8.5.1.1 get_legendre_matrix()

```
gsl::matrix get_legendre_matrix (
            int p,
            int m )
```

8.5.1.2 get_legendre_matrix_inv()

```
gsl::matrix get_legendre_matrix_inv (
            int p,
            int m )
```

8.5.1.3 spheroidal_analysis()

```
gsl::cmatrix spheroidal_analysis (
            gsl::cmatrix f )
```

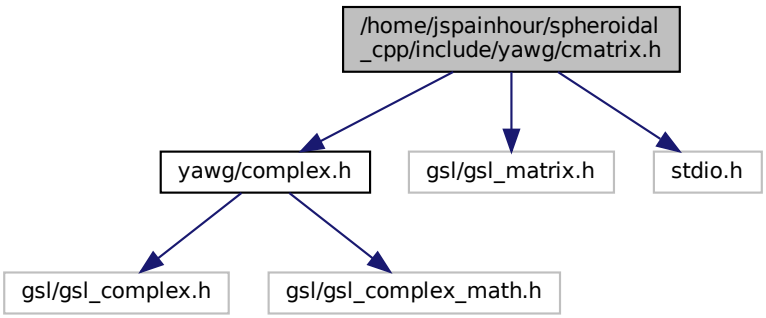
8.5.1.4 spheroidal_snythesis()

```
gsl::cmatrix spheroidal_snythesis (
            gsl::cmatrix shc )
```

8.6 /home/jspainhour/spheroidal_cpp/include/yawg/cmatrix.h File Reference

```
#include <yawg/complex.h>
#include <gsl/gsl_matrix.h>
#include <stdio.h>
```

Include dependency graph for cmatrix.h:



This graph shows which files directly or indirectly include this file:

Classes

- class gsl::cmatrix
- class gsl::cmatrix_view

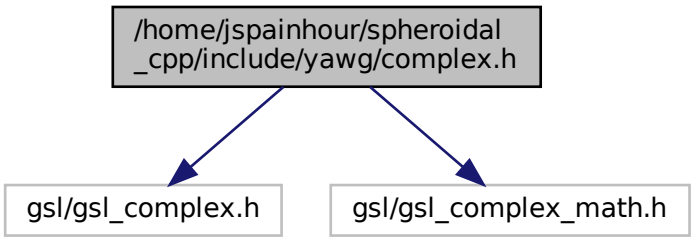
Namespaces

- gsl

8.7 /home/jspainhour/spheroidal_cpp/include/yawg/complex.h File Reference

```
#include <gsl/gsl_complex.h>
#include <gsl/gsl_complex_math.h>
```

Include dependency graph for complex.h:



This graph shows which files directly or indirectly include this file:

Classes

- class gsl::complex

  Wrapper class for gsl_complex structs.
- class gsl::complex_ref

  Stores a refernce to a gsl::complex object.

Namespaces

- gsl
- gsl::complex_literals

Functions

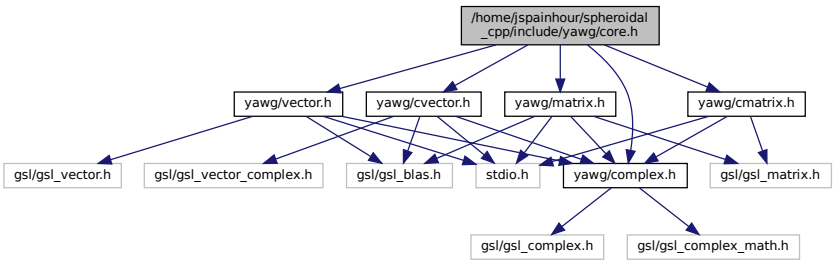
- complex gsl::complex_literals::operator""_i (long double y)

  User defined literal overload for complex numbers.

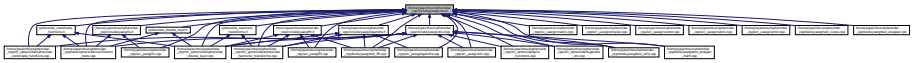
8.8 /home/jspainhour/spheroidal_cpp/include/yawg/core.h File Reference

```
#include <yawg/complex.h>
#include <yawg/vector.h>
#include <yawg/cvector.h>
#include <yawg/matrix.h>
#include <yawg/cmatrix.h>
```

Include dependency graph for core.h:



This graph shows which files directly or indirectly include this file:

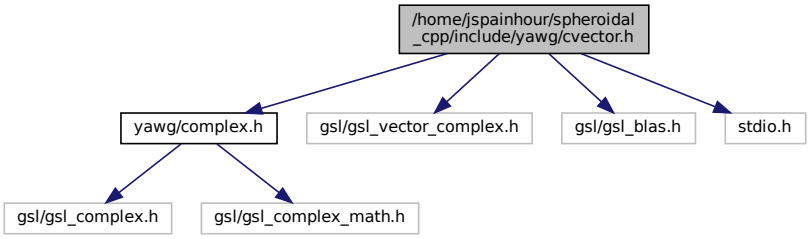


8.9 /home/jspainhour/spheroidal_cpp/include/yawg/cvector.h File Reference

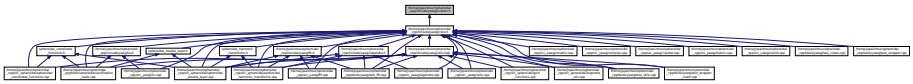
```
#include <yawg/complex.h>
#include <gsl/gsl_vector_complex.h>
#include <gsl/gsl_blas.h>
```

```
#include <stdio.h>
```

Include dependency graph for cvector.h:



This graph shows which files directly or indirectly include this file:



Classes

- class gsl::cvector

  A wrapper class for gsl_vector_complex.
- class gsl::cvector_view
- class gsl::crow_view

  A subclass of cvector_view for stride-1 cvectors.
- class gsl::ccolumn_view

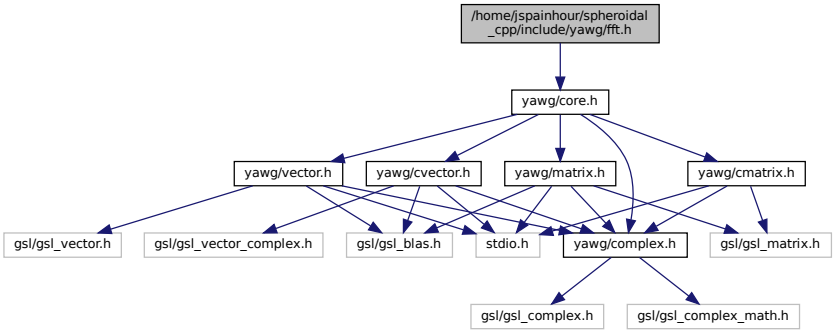
Namespaces

- gsl

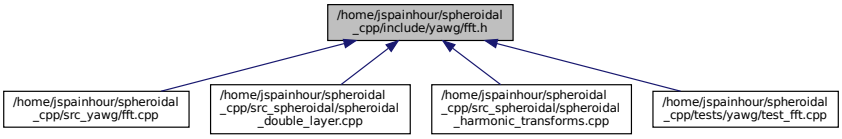
8.10 /home/jspainhour/spheroidal_cpp/include/yawg/fft.h File Reference

```
#include <yawg/core.h>
```

Include dependency graph for fft.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- gsl

Functions

- gsl::cvector gsl::fft (gsl::cvector &&x)

  Compute in-place fft of a gsl::(c)vector.
- gsl::cvector gsl::fft (const gsl::cvector &x)

  Compute fft of a gsl::(c)vector.
- gsl::cvector gsl::ifft (gsl::cvector &&x)

  Compute in-place inverse fft of a gsl::(c)vector.
- gsl::cvector gsl::ifft (const gsl::cvector &x)

  Compute inverse fft of a gsl::(c)vector.
- gsl::cmatrix gsl::fft (gsl::cmatrix &&x, int dim=1)

  Compute in-place 1D fft of each column/row of a gsl::(c)matrix.
- gsl::cmatrix gsl::fft (const gsl::cmatrix &x, int dim=1)

  Compute 1D fft of each column/row of a gsl::(c)matrix.
- gsl::cmatrix gsl::ifft (gsl::cmatrix &&x, int dim=1)

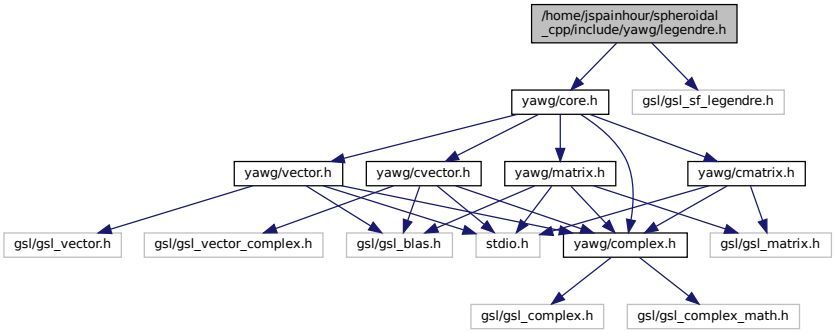
  Compute in-place 1D inverse fft of each column/row of a gsl::(c)matrix.
- gsl::cmatrix gsl::ifft (const gsl::cmatrix &x, int dim=1)

  Compute 1D inverse fft of each column/row of a gsl::(c)matrix.

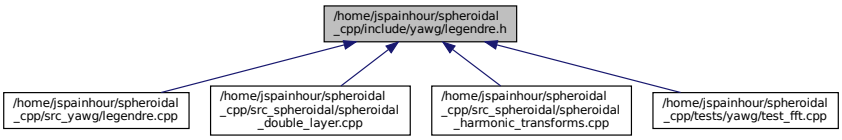
8.11 /home/jspainhour/spheroidal_cpp/include/yawg/legendre.h File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_sf_legendre.h>
```

Include dependency graph for legendre.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- gsl

Enumerations

- enum class gsl::legendre_norm : int { gsl::none = GSL_SF_LEGENDRE_NONE , gsl::schmidt = GSL↩ _SF_LEGENDRE_SCHMIDT , gsl::spharm = GSL_SF_LEGENDRE_SPHARM , gsl::full = GSL_SF_↩ LEGENDRE_FULL }

  Wrapper enum for the GSL gsl_sf_legendre_t.

Functions

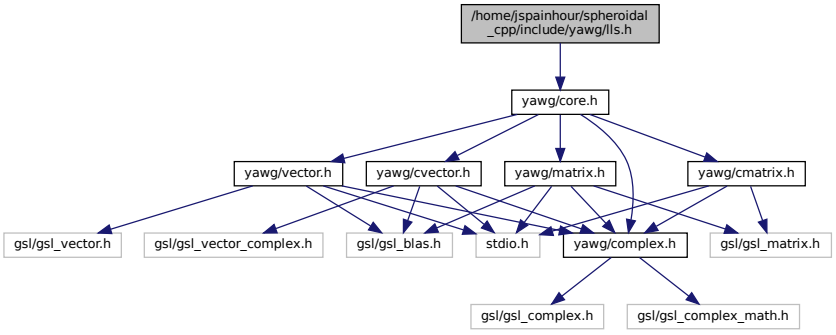
- double gsl::spherical_harmonic (int n, int m, double x)

  Compute the normalized associated Legendre polynomial P_n^m(x)
- complex gsl::spherical_harmonic (int n, int m, double theta, double phi)

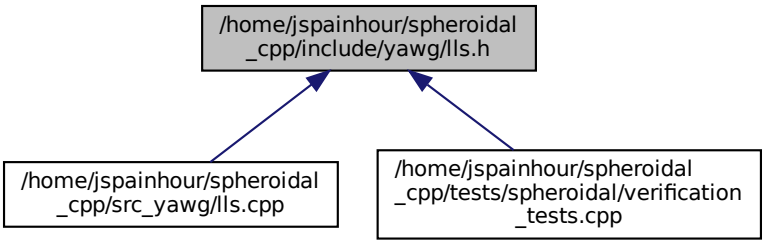
  Compute the normalized associated spherical harmonic Y_n^m(theta, phi)

8.12 /home/jspainhour/spheroidal_cpp/include/yawg/lls.h File Reference

`#include <yawg/core.h>`
Include dependency graph for lls.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- gsl

Functions

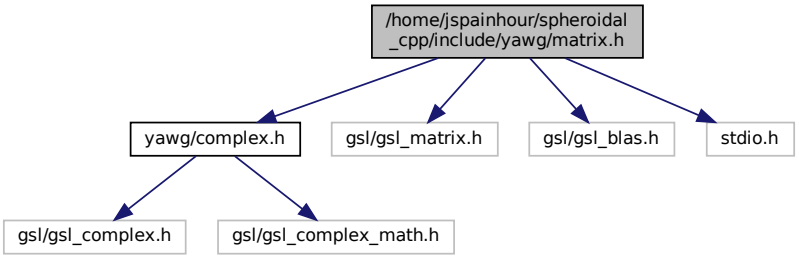
- void gsl::fit_linear (gsl::vector &x, gsl::vector &y, double &c0, double &c1, double &sumsq)

  Compute the intercept `c0` and slope `c1` of best fit

8.13 /home/jspainhour/spheroidal_cpp/include/yawg/matrix.h File Reference

```
#include <yawg/complex.h>
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_blas.h>
#include <stdio.h>
```

Include dependency graph for matrix.h:



This graph shows which files directly or indirectly include this file:

Classes

- class gsl::matrix

  A wrapper class for gsl_matrix.
- class gsl::matrix_view

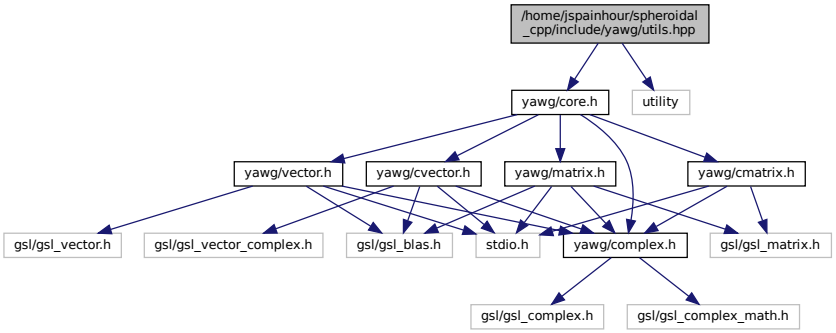
Namespaces

- gsl

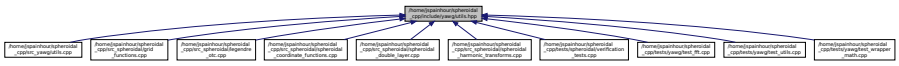
8.14 /home/jspainhour/spheroidal_cpp/include/yawg/utils.hpp File Reference

```
#include <yawg/core.h>
#include <utility>
```

Include dependency graph for utils.hpp:



This graph shows which files directly or indirectly include this file:



Namespaces

- gsl

Functions

- void gsl::leggauss (size_t n, gsl::vector &x, gsl::vector &w, double a=-1.0, double b=1.0)

  Store Gauss-Legendre quadrature nodes and weights.
- vector gsl::leggauss (size_t n, double a=-1.0, double b=1.0)

  Get a vector Gauss-Legendre quadrature nodes.
- gsl::vector gsl::linspace (double a, double b, size_t N=100)

  Get a gsl::vector of evenly spaced points on the interval [a, b] (inclusive)
- gsl::cvector gsl::linspace (gsl::complex a, gsl::complex b, size_t n)

  Complex version of linspace.
- gsl::vector gsl::arange (double a, double b, double step=1.0)

  Get a gsl::vector of `step` spaced points on the interval [`a, b`)
- gsl::vector gsl::circshift (const gsl::vector &x, int k)

  Perform a circular shift of the elements of a gsl::vector.
- gsl::cvector gsl::circshift (const gsl::cvector &x, int k)

  Perform a circular shift of the elements of a gsl::cvector.
- void gsl::meshgrid (const gsl::vector &x, const gsl::vector &y, gsl::matrix &X, gsl::matrix &Y)

  Store 2D grid coordinates based on 1D input gsl::vectors.
- void gsl::meshgrid (const gsl::cvector &x, const gsl::cvector &y, gsl::cmatrix &X, gsl::cmatrix &Y)

  Complex version of gsl::meshgrid.
- gsl::matrix gsl::eye (size_t n)

  Return the nxn identity matrix.
- template<typename Lambda >
  gsl::vector gsl::arrayfun (Lambda &&func, const gsl::vector &x)

  Apply lambda function to each element of a gsl::vector, akin to MATLAB arrayfun.
- template<typename Lambda >
  gsl::vector gsl::arrayfun (Lambda &&func, gsl::vector &&x)

  Move version of arrayfun for vectors.
- template<typename Lambda >
  gsl::cvector gsl::arrayfun (Lambda &&func, const gsl::cvector &x)

  Copy version of arrayfun for complex vectors.
- template<typename Lambda >
  gsl::cvector gsl::arrayfun (Lambda &&func, gsl::cvector &&x)

  Move version of arrayfun for complex vectors.
- template<typename Lambda >
  gsl::matrix gsl::arrayfun (Lambda &&func, const gsl::matrix &x)

  Apply lambda function to each element of a gsl::matrix, akin to MATLAB arrayfun.
- template<typename Lambda >
  gsl::matrix gsl::arrayfun (Lambda &&func, gsl::matrix &&x)

  Move version of arrayfun.
- template<typename Lambda >
  gsl::cmatrix gsl::arrayfun (Lambda &&func, const gsl::cmatrix &x)

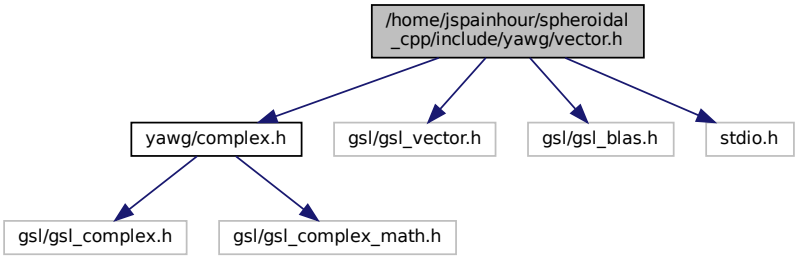
  Copy version of arrayfun for complex matrices.
- template<typename Lambda >
  gsl::cmatrix gsl::arrayfun (Lambda &&func, gsl::cmatrix &&x)

  Move version of arrayfun for complex matrices.
- gsl::vector gsl::diag (const gsl::matrix &A)
- gsl::vector gsl::pow (const gsl::vector &x, double p)
- gsl::cvector gsl::pow (const gsl::cvector &x, complex p)
- gsl::matrix gsl::pow (const gsl::matrix &x, double p)
- gsl::cmatrix gsl::pow (const gsl::cmatrix &x, complex p)

8.15 /home/jspainhour/spheroidal_cpp/include/yawg/vector.h File Reference

```
#include <yawg/complex.h>
#include <gsl/gsl_vector.h>
#include <gsl/gsl_blas.h>
#include <stdio.h>
```

Include dependency graph for vector.h:



This graph shows which files directly or indirectly include this file:

Classes

- class gsl::vector

  A wrapper class for gsl_vector.
- class gsl::vector_view
- class gsl::row_view

  A subclass of cvector_view for stride-1 cvectors.
- class gsl::column_view

  A subclass of vector_view for non-stride-1 vectors.

Namespaces

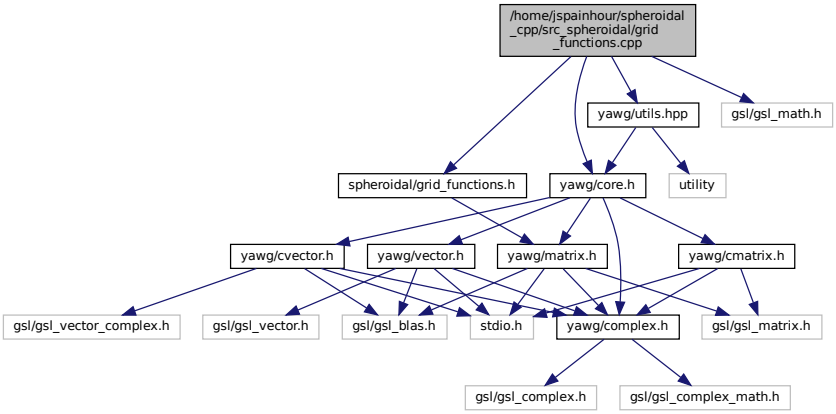
- gsl

8.16 /home/jspainhour/spheroidal_cpp/README.md File Reference

8.17 /home/jspainhour/spheroidal_cpp/src_spheroidal/grid_↩ functions.cpp File Reference

```
#include <spheroidal/grid_functions.h>
#include <yawg/utils.hpp>
#include <yawg/core.h>
#include <gsl/gsl_math.h>
```

Include dependency graph for grid_functions.cpp:



Functions

- int spharm_grid_size_ord (int p, int &nu, int &nv)

  Computes grid size of spheroidal harmonics grid given the order.
- int spharm_grid_size_tot (int ntot, int &nu, int &nv)

  Computes grid size of spheroidal harmonics grid given the total number of points.
- void gl_grid (size_t p, gsl::matrix &U, gsl::matrix &V)

  Populates the matrices `U` and `V` with a spheroidal harmonics grid of order .

8.17.1 Function Documentation

8.17.1.1 gl_grid()

```
void gl_grid (
            size_t p,
            gsl::matrix & U,
            gsl::matrix & V )
```

Populates the matrices `U` and `V` with a spheroidal harmonics grid of order .

Populates the matrices `U` and `V` with a spheroidal harmonics grid of order `p`.

Parameters

p	Order of the spheroidal harmonics
U	Reference to gsl::matrix of Gaussian spaced rows
V	Reference to gsl::matrix of Uniform spaced columns

Mapped to a spheroid, `u` is spaced on [0, pi] and `v` is spaced on [0, 2pi].

8.17.1.2 spharm_grid_size_ord()

```
int spharm_grid_size_ord (
            int p,
            int & nu,
            int & nv )
```

Computes grid size of spheroidal harmonics grid given the order.

Parameters

p	Order of the spheroidal harmonics
nu	Number of points in the theta direction
nv	Number of points in the lambda direction

Returns

The order of the spheroidal harmonics (legacy usage)

8.17.1.3 spharm_grid_size_tot()

```
int spharm_grid_size_tot (
            int ntot,
            int & nu,
            int & nv )
```

Computes grid size of spheroidal harmonics grid given the total number of points.

Parameters

ntot	The total number of points in the grid
nu	Number of points in the theta direction
nv	Number of points in the lambda direction

Note

Will cause an error if the number of points is not valid for a spheroidal harmonics grid

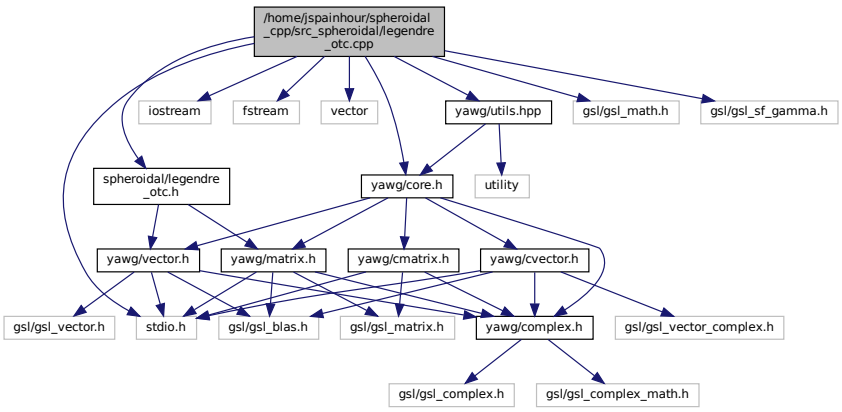
Returns

The order of the spheroidal harmonics (legacy usage)

8.18 /home/jspainhour/spheroidal_cpp/src_spheroidal/legendre_otc.cpp File Reference

```
#include <spheroidal/legendre_otc.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <stdio.h>
#include <yawg/utils.hpp>
#include <yawg/core.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_sf_gamma.h>
```

Include dependency graph for legendre_otc.cpp:



Functions

- gsl::vector cont_frac (int n, int m, gsl::vector u)
- int geti (int n, int m)
- void legendre_otc (int p, gsl::vector u, gsl::matrix &P)

  Computes associated Legendre functions off the cut of the first kind, $$P_n^m(u)$$, $$u>1$$.
- void legendre_otc (int p, gsl::vector u, gsl::matrix &P, gsl::matrix &Q)

  Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, for $$u>1$$.
- void Dlegendre_otc (int p, gsl::vector u, gsl::matrix &P, gsl::matrix &dP)

  Computes associated legendre functions off the cut of the first kind, $$P_n^m(u)$$ and their derivatives.
- void Dlegendre_otc (int p, gsl::vector u, gsl::matrix &P, gsl::matrix &Q, gsl::matrix &dP, gsl::matrix &dQ)

  Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, and their derivatives, for $$u>1$$.

8.18.1 Function Documentation

8.18.1.1 cont_frac()

```
gsl::vector cont_frac (
            int n,
            int m,
            gsl::vector u )
```

8.18.1.2 Dlegendre_otc() [1/2]

```
void Dlegendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P,
            gsl::matrix & dP )
```

Computes associated legendre functions off the cut of the first kind, $$P_n^m(u)$$ and their derivatives.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions
P	Matrix of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
dP	Matrix of derivatives of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

Note

Will cause an error if any u-value is less than or equal to 1.

8.18.1.3 Dlegendre_otc() [2/2]

```
void Dlegendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P,
            gsl::matrix & Q,
            gsl::matrix & dP,
            gsl::matrix & dQ )
```

Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, and their derivatives, for $$u>1$$.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions

Parameters

P	Matrix of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
Q	Matrix of associated legendre functions off the cut of the second kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
dP	Matrix of derivatives of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
dQ	Matrix of derivatives of associated legendre functions off the cut of the second kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

Note

Will cause an error if any u-value is less than or equal to 1.

8.18.1.4 geti()

```
int geti (
            int n,
            int m )
```

8.18.1.5 legendre_otc() [1/2]

```
void legendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P )
```

Computes associated Legendre functions off the cut of the first kind, $$P_n^m(u)$$, $$u>1$$.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions
P	Matrix to store associated Legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

Note

Will cause an error if any u-value is less than or equal to 1.

8.18.1.6 legendre_otc() [2/2]

```
void legendre_otc (
            int p,
            gsl::vector u,
            gsl::matrix & P,
            gsl::matrix & Q )
```

Computes associated legendre functions off the cut of the first and second kind, $$P_n^m(u)$$ and $$Q_n^m(u)$$, for $$u>1$$.

Parameters

p	The order of the spheroidal harmonics
u	The u-values at which to compute the associated legendre functions
P	Matrix of associated legendre functions off the cut of the first kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.
Q	Matrix of associated legendre functions off the cut of the second kind. Each row corresponds to a different n and m value, and each column corresponds to a different u-value.

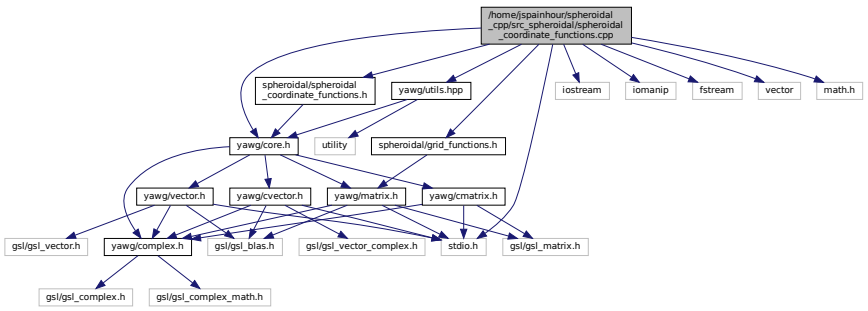
Note

Will cause an error if any u-value is less than or equal to 1.

8.19 /home/jspainhour/spheroidal_cpp/src_spheroidal/spheroidal_coordinate_functions.cpp File Reference

```
#include <spheroidal/spheroidal_coordinate_functions.h>
#include <spheroidal/grid_functions.h>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <vector>
#include <stdio.h>
#include <math.h>
#include <yawg/utils.hpp>
#include <yawg/core.h>
```

Include dependency graph for spheroidal_coordinate_functions.cpp:



Functions

- gsl::matrix spheroidal_to_cart (gsl::matrix S, double a)
- gsl::matrix cart_to_spheroidal (gsl::matrix X, double a)

8.19.1 Function Documentation

8.19.1.1 cart_to_spheroidal()

```
gsl::matrix cart_to_spheroidal (
            gsl::matrix X,
            double a )
```

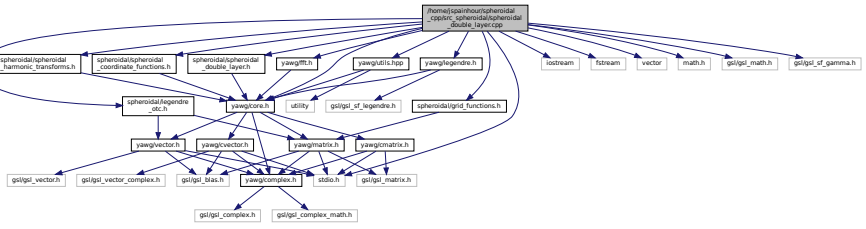
8.19.1.2 spheroidal_to_cart()

```
gsl::matrix spheroidal_to_cart (
            gsl::matrix S,
            double a )
```

8.20 /home/jspainhour/spheroidal_cpp/src_spheroidal/spheroidal_↩ double_layer.cpp File Reference

```
#include <spheroidal/legendre_otc.h>
#include <spheroidal/grid_functions.h>
#include <spheroidal/spheroidal_harmonic_transforms.h>
#include <spheroidal/spheroidal_coordinate_functions.h>
#include <spheroidal/spheroidal_double_layer.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <math.h>
#include <stdio.h>
#include <yawg/utils.hpp>
#include <yawg/core.h>
#include <yawg/fft.h>
#include <yawg/legendre.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_sf_gamma.h>
```

Include dependency graph for spheroidal_double_layer.cpp:



Functions

- gsl::matrix solid_harmonic (int p, gsl::vector u_x, int region)
- void DLspectrum (int p, double u0, gsl::vector &lambda_int, gsl::vector &lambda_surf, gsl::vector &lambda_ext)
- gsl::cmatrix Ynm_matrix (int p, gsl::vector theta, gsl::vector phi)
- gsl::cmatrix spheroidal_double_layer (gsl::cmatrix sigma, double u0, gsl::matrix X, int target_coords)
- gsl::cmatrix spheroidal_double_layer (gsl::cmatrix sigma, double u0)

8.20.1 Function Documentation

8.20.1.1 DLspectrum()

```
void DLspectrum (
            int p,
            double u0,
            gsl::vector & lambda_int,
            gsl::vector & lambda_surf,
            gsl::vector & lambda_ext )
```

8.20.1.2 solid_harmonic()

```
gsl::matrix solid_harmonic (
            int p,
            gsl::vector u_x,
            int region )
```

8.20.1.3 spheroidal_double_layer() [1/2]

```
gsl::cmatrix spheroidal_double_layer (
            gsl::cmatrix sigma,
            double u0 )
```

8.20.1.4 spheroidal_double_layer() [2/2]

```
gsl::cmatrix spheroidal_double_layer (
            gsl::cmatrix sigma,
            double u0,
            gsl::matrix X,
            int target_coords )
```

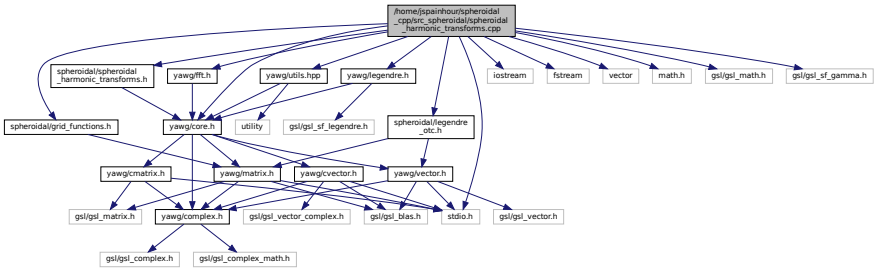
8.20.1.5 Ynm_matrix()

```
gsl::cmatrix Ynm_matrix (
            int p,
            gsl::vector theta,
            gsl::vector phi )
```

8.21 /home/jspainhour/spheroidal_cpp/src_spheroidal/spheroidal_↩ harmonic_transforms.cpp File Reference

```
#include <spheroidal/grid_functions.h>
#include <spheroidal/spheroidal_harmonic_transforms.h>
#include <spheroidal/legendre_otc.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <math.h>
#include <stdio.h>
#include <yawg/utils.hpp>
#include <yawg/core.h>
#include <yawg/fft.h>
#include <yawg/legendre.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_sf_gamma.h>
```

Include dependency graph for spheroidal_harmonic_transforms.cpp:



Functions

- gsl::matrix get_legendre_matrix (int p, int m)
- gsl::matrix get_legendre_matrix_inv (int p, int m)
- gsl::cmatrix spheroidal_analysis (gsl::cmatrix f)
- gsl::cmatrix spheroidal_snythesis (gsl::cmatrix shc)

8.21.1 Function Documentation

8.21.1.1 get_legendre_matrix()

```
gsl::matrix get_legendre_matrix (
            int p,
            int m )
```

8.21.1.2 get_legendre_matrix_inv()

```
gsl::matrix get_legendre_matrix_inv (
            int p,
            int m )
```

8.21.1.3 spheroidal_analysis()

```
gsl::cmatrix spheroidal_analysis (
            gsl::cmatrix f )
```

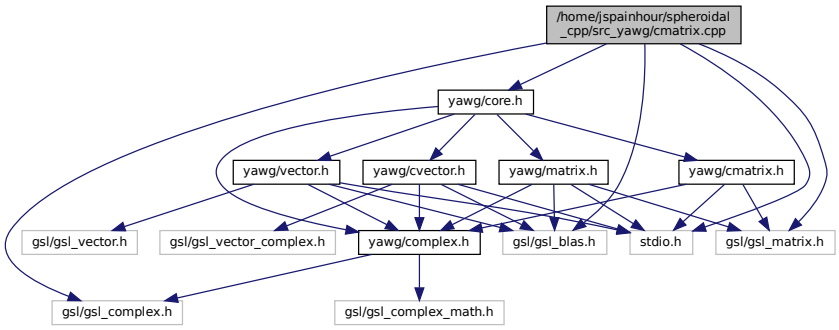
8.21.1.4 spheroidal_snythesis()

```
gsl::cmatrix spheroidal_snythesis (
            gsl::cmatrix shc )
```

8.22 /home/jspainhour/spheroidal_cpp/src_yawg/cmatrix.cpp File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_complex.h>
#include <gsl/gsl_blas.h>
#include <stdio.h>
```

Include dependency graph for cmatrix.cpp:



Namespaces

- gsl

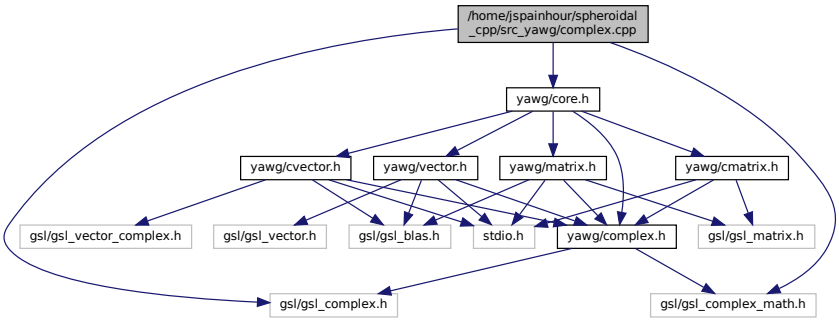
Functions

- cmatrix gsl::operator∗ (double a, const cmatrix &M)
- cmatrix gsl::operator∗ (const cmatrix &M, double a)
- cmatrix gsl::operator∗ (double a, cmatrix &&M)
- cmatrix gsl::operator∗ (cmatrix &&M, double a)
- cmatrix gsl::operator∗ (const cmatrix &M, const complex &a)
- cmatrix gsl::operator∗ (const complex &a, const cmatrix &M)
- cmatrix gsl::operator∗ (complex z, const cmatrix &M)
- cmatrix gsl::operator∗ (const cmatrix &M, complex z)
- cmatrix gsl::operator∗ (cmatrix &&M, complex z)
- cmatrix gsl::operator∗ (complex z, cmatrix &&M)
- cmatrix gsl::operator∗ (complex a, const matrix &M)
- cmatrix gsl::operator∗ (const matrix &M, complex a)
- cmatrix gsl::operator/ (double a, const cmatrix &M)
- cmatrix gsl::operator/ (const cmatrix &M, double a)
- cmatrix gsl::operator/ (double a, cmatrix &&M)
- cmatrix gsl::operator/ (cmatrix &&M, double a)
- cmatrix gsl::operator/ (const cmatrix &M, const complex &a)
- cmatrix gsl::operator/ (const complex &a, const cmatrix &M)
- cmatrix gsl::operator/ (complex z, const cmatrix &M)
- cmatrix gsl::operator/ (const cmatrix &M, complex z)
- cmatrix gsl::operator/ (cmatrix &&M, complex z)
- cmatrix gsl::operator/ (complex z, cmatrix &&M)
- cmatrix gsl::operator/ (complex z, const matrix &M)
- cmatrix gsl::operator/ (const matrix &M, complex z)
- cmatrix gsl::operator+ (const cmatrix &M1, const cmatrix &M2)
- cmatrix gsl::operator+ (cmatrix &&M1, const cmatrix &M2)
- cmatrix gsl::operator+ (const cmatrix &M1, cmatrix &&M2)
- cmatrix gsl::operator+ (cmatrix &&M1, cmatrix &&M2)
- cmatrix gsl::operator+ (const cmatrix &M1, const matrix &M2)
- cmatrix gsl::operator+ (cmatrix &&M1, const matrix &M2)
- cmatrix gsl::operator+ (const matrix &M1, const cmatrix &M2)
- cmatrix gsl::operator+ (const matrix &M1, cmatrix &&M2)
- cmatrix gsl::operator- (const cmatrix &M1, const cmatrix &M2)
- cmatrix gsl::operator- (cmatrix &&M1, const cmatrix &M2)
- cmatrix gsl::operator- (const cmatrix &M1, cmatrix &&M2)
- cmatrix gsl::operator- (cmatrix &&M1, cmatrix &&M2)
- cmatrix gsl::operator- (const cmatrix &M1, const matrix &M2)
- cmatrix gsl::operator- (cmatrix &&M1, const matrix &M2)
- cmatrix gsl::operator- (const matrix &M1, const cmatrix &M2)
- cmatrix gsl::operator- (const matrix &M1, cmatrix &&M2)
- cmatrix gsl::operator∗ (const cmatrix &A, const cmatrix &B)
- bool gsl::operator== (const cmatrix &M1, const cmatrix &M2)
- bool gsl::operator!= (const cmatrix &M1, const cmatrix &M2)
- bool gsl::operator== (const cmatrix &M1, const matrix &M2)
- bool gsl::operator!= (const cmatrix &M1, const matrix &M2)
- bool gsl::operator== (const matrix &M1, const cmatrix &M2)
- bool gsl::operator!= (const matrix &M1, const cmatrix &M2)

8.23 /home/jspainhour/spheroidal_cpp/src_yawg/complex.cpp File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_complex.h>
#include <gsl/gsl_complex_math.h>
```

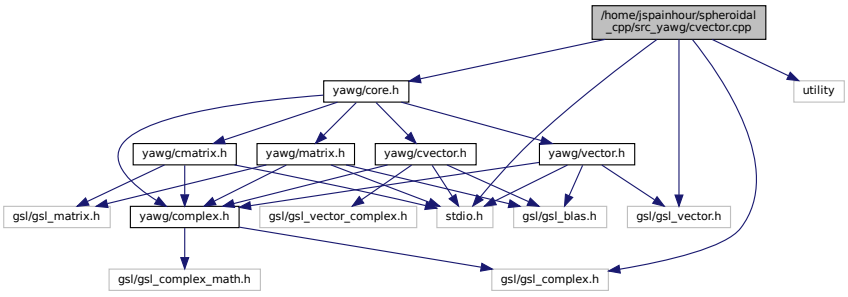
Include dependency graph for complex.cpp:



8.24 /home/jspainhour/spheroidal_cpp/src_yawg/cvector.cpp File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_vector.h>
#include <gsl/gsl_complex.h>
#include <stdio.h>
#include <utility>
```

Include dependency graph for cvector.cpp:



Namespaces

- gsl

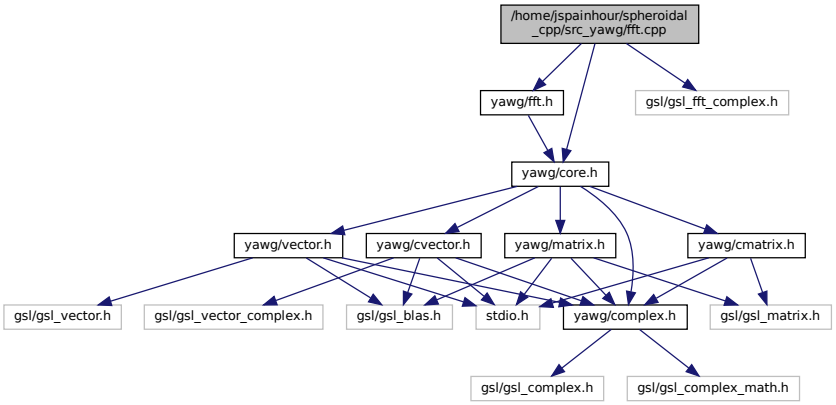
Functions

- cvector gsl::operator∗ (complex z, const cvector &v)
- cvector gsl::operator∗ (complex z, cvector &&v)
- cvector gsl::operator∗ (const cvector &v, complex z)
- cvector gsl::operator∗ (cvector &&v, complex z)
- cvector gsl::operator∗ (complex a, const vector &v)
- cvector gsl::operator∗ (const vector &v, complex a)
- cvector gsl::operator∗ (const cvector &v, double x)
- cvector gsl::operator∗ (double x, const cvector &v)
- cvector gsl::operator∗ (cvector &&v, double x)
- cvector gsl::operator∗ (double x, cvector &&v)
- cvector gsl::operator/ (const cvector &v, complex z)
- cvector gsl::operator/ (cvector &&v, complex z)
- cvector gsl::operator/ (complex z, const cvector &v)
- cvector gsl::operator/ (complex z, cvector &&v)
- cvector gsl::operator/ (const vector &v, complex z)
- cvector gsl::operator/ (complex z, const vector &v)
- cvector gsl::operator/ (const cvector &v, double x)
- cvector gsl::operator/ (cvector &&v, double x)
- cvector gsl::operator/ (double x, const cvector &v)
- cvector gsl::operator/ (double x, cvector &&v)
- cvector gsl::operator+ (const cvector &v1, const cvector &v2)
- cvector gsl::operator+ (cvector &&v1, const cvector &v2)
- cvector gsl::operator+ (const cvector &v1, cvector &&v2)
- cvector gsl::operator+ (cvector &&v1, cvector &&v2)
- cvector gsl::operator- (const cvector &v1, const cvector &v2)
- cvector gsl::operator- (cvector &&v1, const cvector &v2)
- cvector gsl::operator- (const cvector &v1, cvector &&v2)
- cvector gsl::operator- (cvector &&v1, cvector &&v2)
- cvector gsl::operator- (const vector &v1, const cvector &v2)
- cvector gsl::operator- (const vector &v1, cvector &&v2)
- cvector gsl::operator- (const cvector &v1, const vector &v2)
- cvector gsl::operator- (cvector &&v1, const vector &v2)
- bool gsl::operator== (const cvector &v1, const cvector &v2)
- bool gsl::operator!= (const cvector &v1, const cvector &v2)
- bool gsl::operator== (const vector &v1, const cvector &v2)
- bool gsl::operator!= (const vector &v1, const cvector &v2)
- bool gsl::operator== (const cvector &v1, const vector &v2)
- bool gsl::operator!= (const cvector &v1, const vector &v2)
- cvector gsl::operator∗ (const cmatrix &M, const cvector &v)

8.25 /home/jspainhour/spheroidal_cpp/src_yawg/fft.cpp File Reference

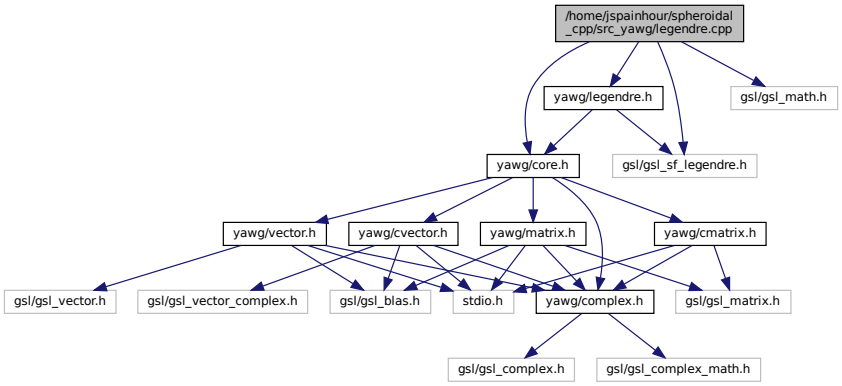
```
#include <yawg/core.h>
#include <yawg/fft.h>
```

```
#include <gsl/gsl_fft_complex.h>
```
Include dependency graph for fft.cpp:



8.26 /home/jspainhour/spheroidal_cpp/src_yawg/legendre.cpp File Reference

```
#include <yawg/core.h>
#include <yawg/legendre.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_sf_legendre.h>
```
Include dependency graph for legendre.cpp:

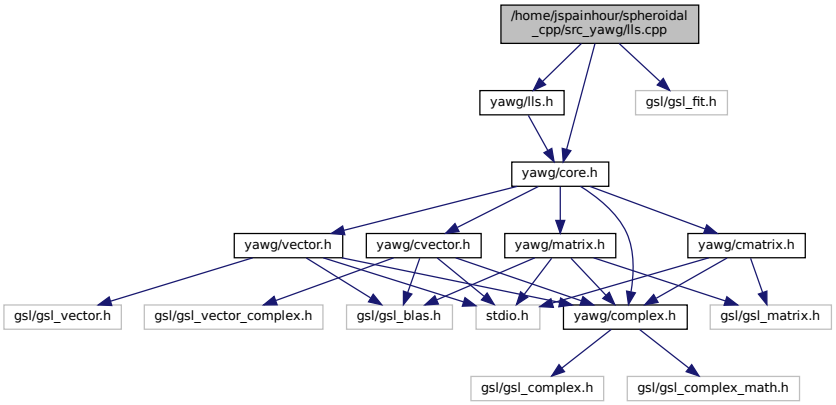


8.27 /home/jspainhour/spheroidal_cpp/src_yawg/lls.cpp File Reference

```
#include <yawg/core.h>
#include <yawg/lls.h>
```

```
#include <gsl/gsl_fit.h>
```

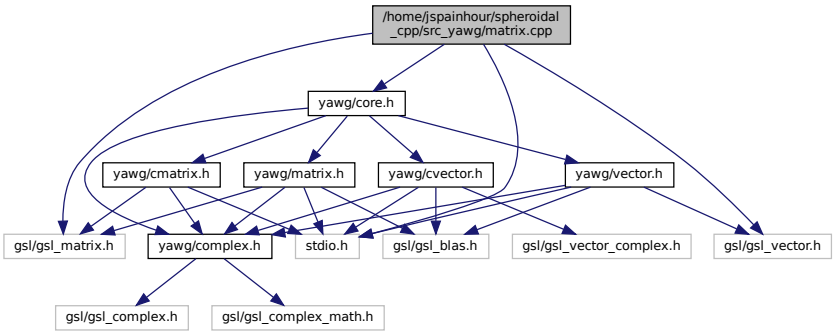
Include dependency graph for lls.cpp:



8.28 /home/jspainhour/spheroidal_cpp/src_yawg/matrix.cpp File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_vector.h>
#include <gsl/gsl_matrix.h>
#include <stdio.h>
```

Include dependency graph for matrix.cpp:



Namespaces

- gsl

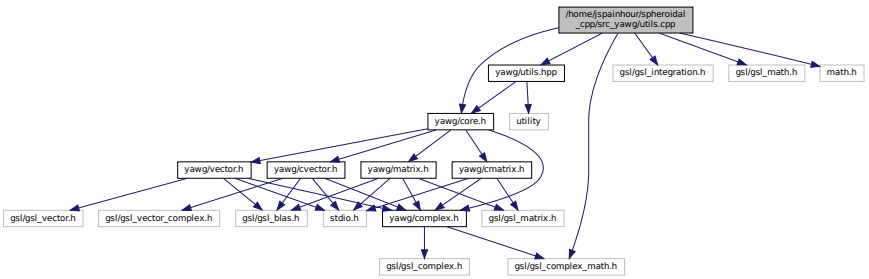
Functions

- matrix gsl::operator∗ (double a, const matrix &M)
- matrix gsl::operator∗ (double a, matrix &&M)
- matrix gsl::operator∗ (const matrix &M, double a)
- matrix gsl::operator∗ (matrix &&M, double a)
- matrix gsl::operator/ (double a, const matrix &M)
- matrix gsl::operator/ (double a, matrix &&M)
- matrix gsl::operator/ (const matrix &M, double a)
- matrix gsl::operator/ (matrix &&M, double a)
- matrix gsl::operator+ (const matrix &M1, const matrix &M2)
- matrix gsl::operator+ (const matrix &M1, matrix &&M2)
- matrix gsl::operator+ (matrix &&M1, const matrix &M2)
- matrix gsl::operator+ (matrix &&M1, matrix &&M2)
- matrix gsl::operator- (const matrix &M1, const matrix &M2)
- matrix gsl::operator- (const matrix &M1, matrix &&M2)
- matrix gsl::operator- (matrix &&M1, const matrix &M2)
- matrix gsl::operator- (matrix &&M1, matrix &&M2)
- matrix gsl::operator∗ (const matrix &A, const matrix &B)
- bool gsl::operator== (const matrix &M1, const matrix &M2)
- bool gsl::operator!= (const matrix &M1, const matrix &M2)

8.29 /home/jspainhour/spheroidal_cpp/src_yawg/utils.cpp File Reference

```
#include <yawg/core.h>
#include <yawg/utils.hpp>
#include <gsl/gsl_integration.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_complex_math.h>
#include <math.h>
```

Include dependency graph for utils.cpp:

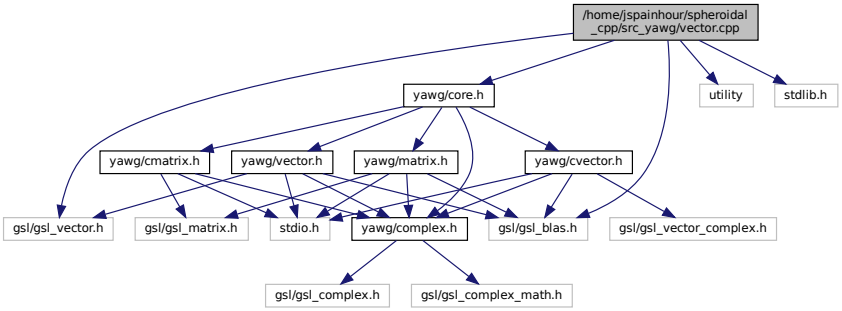


8.30 /home/jspainhour/spheroidal_cpp/src_yawg/vector.cpp File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_vector.h>
```

```
#include <gsl/gsl_blas.h>
#include <utility>
#include <stdlib.h>
```

Include dependency graph for vector.cpp:



Namespaces

- gsl

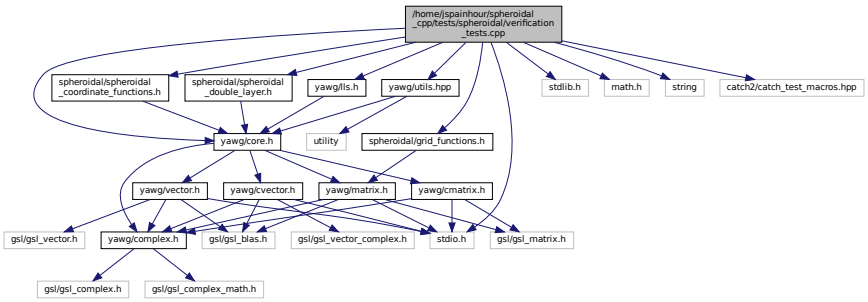
Functions

- vector gsl::operator∗ (double a, const vector &v)
- vector gsl::operator∗ (double a, vector &&v)
- vector gsl::operator∗ (const vector &v, double a)
- vector gsl::operator∗ (vector &&v, double a)
- vector gsl::operator/ (double a, const vector &v)
- vector gsl::operator/ (double a, vector &&v)
- vector gsl::operator/ (const vector &v, double a)
- vector gsl::operator/ (vector &&v, double a)
- vector gsl::operator+ (const vector &v1, const vector &v2)
- vector gsl::operator+ (vector &&v1, const vector &v2)
- vector gsl::operator+ (const vector &v1, vector &&v2)
- vector gsl::operator+ (vector &&v1, vector &&v2)
- cvector gsl::operator+ (const vector &v1, const cvector &v2)
- cvector gsl::operator+ (const vector &v1, cvector &&v2)
- cvector gsl::operator+ (const cvector &v1, const vector &v2)
- cvector gsl::operator+ (cvector &&v1, const vector &v2)
- vector gsl::operator- (const vector &v1, const vector &v2)
- vector gsl::operator- (vector &&v1, const vector &v2)
- vector gsl::operator- (const vector &v1, vector &&v2)
- vector gsl::operator- (vector &&v1, vector &&v2)
- bool gsl::operator== (const vector &v1, const vector &v2)
- bool gsl::operator!= (const vector &v1, const vector &v2)
- vector gsl::operator∗ (const matrix &M, const vector &v)

8.31 /home/jspainhour/spheroidal_cpp/tests/spheroidal/verification_tests.cpp File Reference

```
#include <spheroidal/spheroidal_coordinate_functions.h>
#include <spheroidal/spheroidal_double_layer.h>
#include <spheroidal/grid_functions.h>
#include <yawg/utils.hpp>
#include <yawg/core.h>
#include <yawg/lls.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string>
#include <catch2/catch_test_macros.hpp>
```

Include dependency graph for verification_tests.cpp:



Macros

- #define HW6_PLOTS
- #define CATCH_CONFIG_MAIN

Functions

- gsl::vector PtChargePotential (gsl::vector ptch, gsl::matrix Xptch, gsl::matrix Y)
- gsl::matrix test_density (int p)
- TEST_CASE ("Convergence Testing", "[spheroidal_double_layer]")

  Perform a convergence test for the spheroidal_double_layer function.

8.31.1 Macro Definition Documentation

8.31.1.1 CATCH_CONFIG_MAIN

```
#define CATCH_CONFIG_MAIN
```

8.31.1.2 HW6_PLOTS

```
#define HW6_PLOTS
```

8.31.2 Function Documentation

8.31.2.1 PtChargePotential()

```
gsl::vector PtChargePotential (
            gsl::vector ptch,
            gsl::matrix Xptch,
            gsl::matrix Y )
```

8.31.2.2 TEST_CASE()

```
TEST_CASE (
            "Convergence Testing" ,
            "" [spheroidal_double_layer] )
```

Perform a convergence test for the spheroidal_double_layer function.

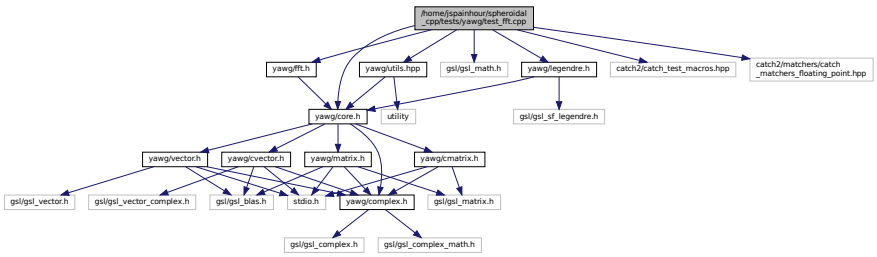
8.31.2.3 test_density()

```
gsl::matrix test_density (
            int p )
```

8.32 /home/jspainhour/spheroidal_cpp/tests/yawg/test_fft.cpp File Reference

```
#include <yawg/core.h>
#include <yawg/utils.hpp>
#include <yawg/fft.h>
#include <gsl/gsl_math.h>
#include <yawg/legendre.h>
#include <catch2/catch_test_macros.hpp>
#include <catch2/matchers/catch_matchers_floating_point.hpp>
```

Include dependency graph for test_fft.cpp:



Macros

- #define CATCH_CONFIG_MAIN

Functions

- TEST_CASE ("gsl::fft", "[gsl::fft]")
- TEST_CASE ("gsl::fft with non-power-of-2 input size", "[gsl::fft]")
- TEST_CASE ("gsl::fft on 3x4 matrix", "[gsl::fft]")
- TEST_CASE ("gsl::ifft on 3x4 matrix", "[gsl::ifft]")
- TEST_CASE ("legendre_P", "[legendre_P]")

8.32.1 Macro Definition Documentation

8.32.1.1 CATCH_CONFIG_MAIN

```
#define CATCH_CONFIG_MAIN
```

8.32.2 Function Documentation

8.32.2.1 TEST_CASE() [1/5]

```
TEST_CASE (
            "gsl::fft on 3x4 matrix" ,
            "" [gsl::fft] )
```

8.32.2.2 TEST_CASE() [2/5]

```
TEST_CASE (
            "gsl::fft with non-power-of-2 input size" ,
            "" [gsl::fft] )
```

8.32.2.3 TEST_CASE() [3/5]

```
TEST_CASE (
            "gsl::fft" ,
            "" [gsl::fft] )
```

8.32.2.4 TEST_CASE() [4/5]

```
TEST_CASE (
            "gsl::ifft on 3x4 matrix" ,
            "" [gsl::ifft] )
```

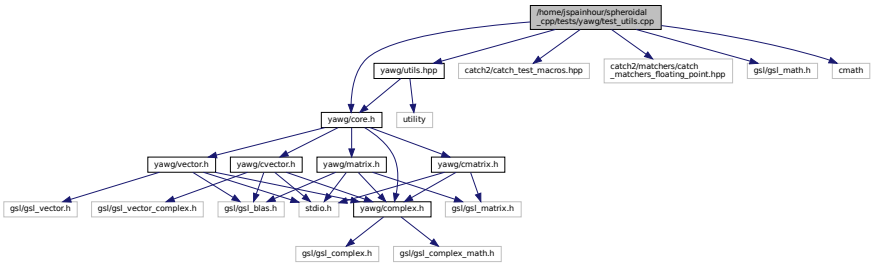
8.32.2.5 TEST_CASE() [5/5]

```
TEST_CASE (
            "legendre_P" ,
            "" [legendre_P] )
```

8.33 /home/jspainhour/spheroidal_cpp/tests/yawg/test_utils.cpp File Reference

```
#include <yawg/core.h>
#include <yawg/utils.hpp>
#include <catch2/catch_test_macros.hpp>
#include <catch2/matchers/catch_matchers_floating_point.hpp>
#include <gsl/gsl_math.h>
#include <cmath>
```

Include dependency graph for test_utils.cpp:



Macros

- #define CATCH_CONFIG_MAIN

Functions

- TEST_CASE ("linspace", "[linspace]")
- TEST_CASE ("arange", "[arange]")
- TEST_CASE ("leggauss", "[leggauss]")
- TEST_CASE ("leggauss return", "[leggauss]")
- TEST_CASE ("meshgrid", "[meshgrid]")
- TEST_CASE ("eye", "[eye]")
- TEST_CASE ("arrayfun vector", "[arrayfun]")
- TEST_CASE ("arrayfun matrix", "[arrayfun]")
- TEST_CASE ("circshift vector", "[circshift]")

8.33.1 Macro Definition Documentation

8.33.1.1 CATCH_CONFIG_MAIN

```
#define CATCH_CONFIG_MAIN
```

8.33.2 Function Documentation

8.33.2.1 TEST_CASE() [1/9]

```
TEST_CASE (
            "arange" ,
            "" [arange] )
```

8.33.2.2 TEST_CASE() [2/9]

```
TEST_CASE (
            "arrayfun matrix" ,
            "" [arrayfun] )
```

8.33.2.3 TEST_CASE() [3/9]

```
TEST_CASE (
            "arrayfun vector" ,
            "" [arrayfun] )
```

8.33.2.4 TEST_CASE() [4/9]

```
TEST_CASE (
            "circshift vector" ,
            "" [circshift] )
```

8.33.2.5 TEST_CASE() [5/9]

```
TEST_CASE (
            "eye" ,
            "" [eye] )
```

8.33.2.6 TEST_CASE() [6/9]

```
TEST_CASE (
            "leggauss return" ,
            "" [leggauss] )
```

8.33.2.7 TEST_CASE() [7/9]

```
TEST_CASE (
            "leggauss" ,
            "" [leggauss] )
```

8.33.2.8 TEST_CASE() [8/9]

```
TEST_CASE (
            "linspace" ,
            "" [linspace] )
```

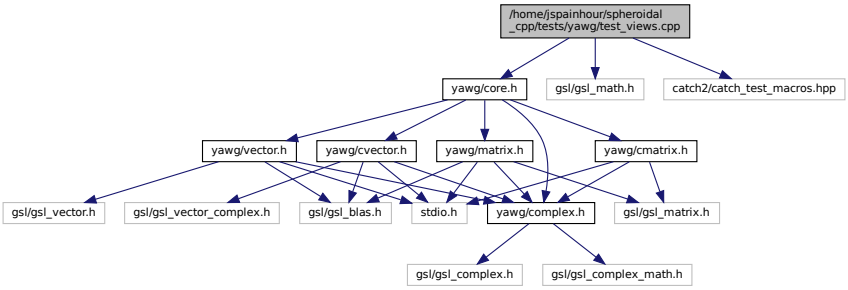
8.33.2.9 TEST_CASE() [9/9]

```
TEST_CASE (
            "meshgrid" ,
            "" [meshgrid] )
```

8.34 /home/jspainhour/spheroidal_cpp/tests/yawg/test_views.cpp File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_math.h>
#include <catch2/catch_test_macros.hpp>
```

Include dependency graph for test_views.cpp:



Macros

- #define CATCH_CONFIG_MAIN

Functions

- TEST_CASE ("gsl::vector_view operations", "[gsl::vector_view]")
- TEST_CASE ("gsl::cvector operations", "[gsl::cvector]")
- TEST_CASE ("matrix_view", "[matrix_view]")
- TEST_CASE ("cmatrix_view", "[cmatrix_view]")
- TEST_CASE ("gsl::matrix column and row view methods", "[gsl::matrix][gsl::row_view][gsl::column_view]")
- TEST_CASE ("gsl::cmatrix column and row view methods", "[gsl::cmatrix][gsl::crow_view][gsl::ccolumn_view]")

8.34.1 Macro Definition Documentation

8.34.1.1 CATCH_CONFIG_MAIN

```
#define CATCH_CONFIG_MAIN
```

8.34.2 Function Documentation

8.34.2.1 TEST_CASE() [1/6]

```
TEST_CASE (
            "cmatrix_view" ,
            "" [cmatrix_view] )
```

8.34.2.2 TEST_CASE() [2/6]

```
TEST_CASE (
            "gsl::cmatrix column and row view methods" ,
            "" [gsl::cmatrix][gsl::crow_view][gsl::ccolumn_view] )
```

8.34.2.3 TEST_CASE() [3/6]

```
TEST_CASE (
            "gsl::cvector operations" ,
            "" [gsl::cvector] )
```

8.34.2.4 TEST_CASE() [4/6]

```
TEST_CASE (
            "gsl::matrix column and row view methods" ,
            "" [gsl::matrix][gsl::row_view][gsl::column_view] )
```

8.34.2.5 TEST_CASE() [5/6]

```
TEST_CASE (
            "gsl::vector_view operations" ,
            "" [gsl::vector_view] )
```

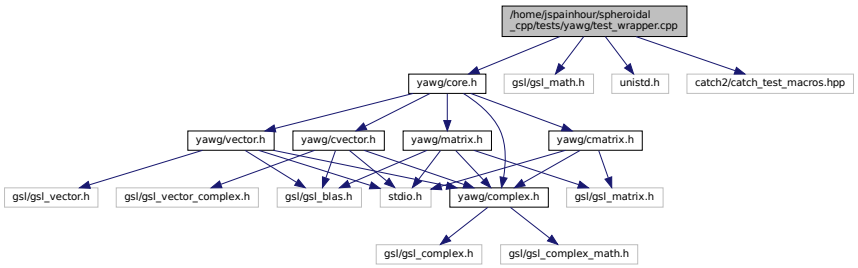
8.34.2.6 TEST_CASE() [6/6]

```
TEST_CASE (
            "matrix_view" ,
            "" [matrix_view] )
```

8.35 /home/jspainhour/spheroidal_cpp/tests/yawg/test_wrapper.cpp File Reference

```
#include <yawg/core.h>
#include <gsl/gsl_math.h>
#include <unistd.h>
#include <catch2/catch_test_macros.hpp>
```

Include dependency graph for test_wrapper.cpp:



Macros

- #define CATCH_CONFIG_MAIN

Functions

- TEST_CASE ("gsl::vector constructors", "[gsl::vector]")
- TEST_CASE ("gsl::complex constructors", "[gsl::complex]")
- TEST_CASE ("gsl::cvector constructors", "[gsl::cvector]")
- TEST_CASE ("gsl::vector -> gsl::cvector conversion", "[gsl::vector][gsl::cvector]")
- TEST_CASE ("gsl::matrix -> gsl::cmatrix conversion", "[gsl::matrix][gsl::cmatrix]")
- TEST_CASE ("gsl::vector assignment operators", "[gsl::vector]")
- TEST_CASE ("gsl::cvector assignment operators", "[gsl::cvector]")
- TEST_CASE ("gsl::vector resize", "[gsl::vector]")
- TEST_CASE ("gsl::cvector resize", "[gsl::cvector]")
- TEST_CASE ("gsl::vector element access", "[gsl::vector]")
- TEST_CASE ("gsl::cvector element access", "[gsl::cvector]")
- TEST_CASE ("gsl::vector print", "[gsl::vector][gsl::cector][gsl::matrix][gsl::cmatrix]")
- TEST_CASE ("gsl::matrix save_csv and load_csv", "[gsl::matrix]")
- TEST_CASE ("gsl::cmatrix save_csv and load_csv", "[gsl::cmatrix]")
- TEST_CASE ("gsl::matrix constructors", "[gsl::matrix]")
- TEST_CASE ("gsl::cmatrix constructors", "[gsl::cmatrix]")
- TEST_CASE ("gsl::matrix assignment", "[gsl::matrix]")
- TEST_CASE ("gsl::cmatrix assignment operators", "[gsl::cmatrix]")
- TEST_CASE ("gsl::matrix element access", "[gsl::matrix]")
- TEST_CASE ("gsl::cmatrix element access", "[gsl::cmatrix]")
- TEST_CASE ("gsl::matrix resize", "[gsl::matrix]")
- TEST_CASE ("gsl::cmatrix resize", "[gsl::cmatrix]")
- TEST_CASE ("gsl::matrix reshape", "[gsl::matrix]")
- TEST_CASE ("gsl::cmatrix reshape", "[gsl::cmatrix]")

8.35.1 Macro Definition Documentation

8.35.1.1 CATCH_CONFIG_MAIN

```
#define CATCH_CONFIG_MAIN
```

8.35.2 Function Documentation

8.35.2.1 TEST_CASE() [1/24]

```
TEST_CASE (
            "gsl::cmatrix assignment operators" ,
            "" [gsl::cmatrix] )
```

8.35.2.2 TEST_CASE() [2/24]

```
TEST_CASE (
            "gsl::cmatrix constructors" ,
            "" [gsl::cmatrix] )
```

8.35.2.3 TEST_CASE() [3/24]

```
TEST_CASE (
            "gsl::cmatrix element access" ,
            "" [gsl::cmatrix] )
```

8.35.2.4 TEST_CASE() [4/24]

```
TEST_CASE (
            "gsl::cmatrix reshape" ,
            "" [gsl::cmatrix] )
```

8.35.2.5 TEST_CASE() [5/24]

```
TEST_CASE (
            "gsl::cmatrix resize" ,
            "" [gsl::cmatrix] )
```

8.35.2.6 TEST_CASE() [6/24]

```
TEST_CASE (
            "gsl::cmatrix save_csv and load_csv" ,
            "" [gsl::cmatrix] )
```

8.35.2.7 TEST_CASE() [7/24]

```
TEST_CASE (
            "gsl::complex constructors" ,
            "" [gsl::complex] )
```

8.35.2.8 TEST_CASE() [8/24]

```
TEST_CASE (
            "gsl::cvector assignment operators" ,
            "" [gsl::cvector] )
```

8.35.2.9 TEST_CASE() [9/24]

```
TEST_CASE (
            "gsl::cvector constructors" ,
            "" [gsl::cvector] )
```

8.35.2.10 TEST_CASE() [10/24]

```
TEST_CASE (
            "gsl::cvector element access" ,
            "" [gsl::cvector] )
```

8.35.2.11 TEST_CASE() [11/24]

```
TEST_CASE (
            "gsl::cvector resize" ,
            "" [gsl::cvector] )
```

8.35.2.12 TEST_CASE() [12/24]

```
TEST_CASE (
            "gsl::matrix -> gsl::cmatrix conversion" ,
            "" [gsl::matrix][gsl::cmatrix] )
```

8.35.2.13 TEST_CASE() [13/24]

```
TEST_CASE (
            "gsl::matrix assignment" ,
            "" [gsl::matrix] )
```

8.35.2.14 TEST_CASE() [14/24]

```
TEST_CASE (
            "gsl::matrix constructors" ,
            "" [gsl::matrix] )
```

8.35.2.15 TEST_CASE() [15/24]

```
TEST_CASE (
            "gsl::matrix element access" ,
            "" [gsl::matrix] )
```

8.35.2.16 TEST_CASE() [16/24]

```
TEST_CASE (
            "gsl::matrix reshape" ,
            "" [gsl::matrix] )
```

8.35.2.17 TEST_CASE() [17/24]

```
TEST_CASE (
            "gsl::matrix resize" ,
            "" [gsl::matrix] )
```

8.35.2.18 TEST_CASE() [18/24]

```
TEST_CASE (
            "gsl::matrix save_csv and load_csv" ,
            "" [gsl::matrix] )
```

8.35.2.19 TEST_CASE() [19/24]

```
TEST_CASE (
            "gsl::vector -> gsl::cvector conversion" ,
            "" [gsl::vector][gsl::cvector] )
```

8.35.2.20 TEST_CASE() [20/24]

```
TEST_CASE (
            "gsl::vector assignment operators" ,
            "" [gsl::vector] )
```

8.35.2.21 TEST_CASE() [21/24]

```
TEST_CASE (
            "gsl::vector constructors" ,
            "" [gsl::vector] )
```

8.35.2.22 TEST_CASE() [22/24]

```
TEST_CASE (
            "gsl::vector element access" ,
            "" [gsl::vector] )
```

8.35.2.23 TEST_CASE() [23/24]

```
TEST_CASE (
            "gsl::vector print" ,
            "" [gsl::vector][gsl::cector][gsl::matrix][gsl::cmatrix] )
```

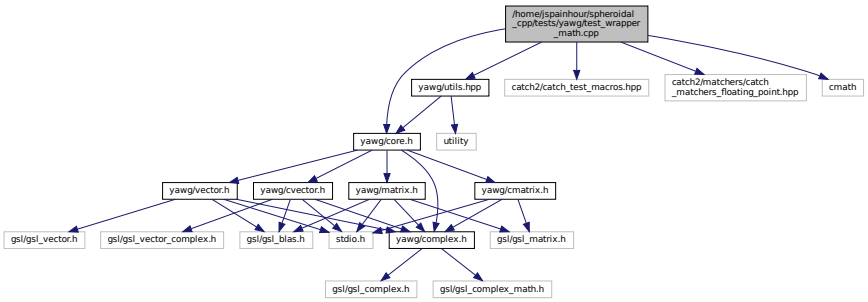
8.35.2.24 TEST_CASE() [24/24]

```
TEST_CASE (
            "gsl::vector resize" ,
            "" [gsl::vector] )
```

8.36 /home/jspainhour/spheroidal_cpp/tests/yawg/test_wrapper_math.cpp File Reference

```
#include <yawg/core.h>
#include <yawg/utils.hpp>
#include <catch2/catch_test_macros.hpp>
#include <catch2/matchers/catch_matchers_floating_point.hpp>
#include <cmath>
```

Include dependency graph for test_wrapper_math.cpp:



Macros

- #define CATCH_CONFIG_MAIN

Functions

- TEST_CASE ("gsl::complex methods", "[gsl::complex]")
- TEST_CASE ("gsl::complex assignment operators", "[gsl::complex]")
- TEST_CASE ("gsl::complex operators", "[gsl::complex]")
- TEST_CASE ("gsl::vector addition", "[gsl::vector]")

  Use Catch2 to test addition of gsl::vectors.
- TEST_CASE ("gsl::vector and gsl::cvector addition", "[gsl::vector][gsl::cvector]")

  Use Catch2 to test addition of gsl::vectors and gsl::cvectors.
- TEST_CASE ("gsl::vector and gsl::cvector scalar multiplication", "[gsl::vector][gsl::cvector]")

  Use Catch2 to test scalar multiplication of gsl::vectors and gsl::cvectors.
- TEST_CASE ("gsl::vector +=, -=, ∗=, /=", "[gsl::vector]")

  Use Catch2 to test +=, -=, ∗=, /= of gsl::vectors.

8.36.1 Macro Definition Documentation

8.36.1.1 CATCH_CONFIG_MAIN

```
#define CATCH_CONFIG_MAIN
```

8.36.2 Function Documentation

8.36.2.1 TEST_CASE() [1/7]

```
TEST_CASE (
            "gsl::complex assignment operators" ,
            "" [gsl::complex] )
```

8.36.2.2 TEST_CASE() [2/7]

```
TEST_CASE (
            "gsl::complex methods" ,
            "" [gsl::complex] )
```

8.36.2.3 TEST_CASE() [3/7]

```
TEST_CASE (
            "gsl::complex operators" ,
            "" [gsl::complex] )
```

8.36.2.4 TEST_CASE() [4/7]

```
TEST_CASE (
            "gsl::vector + ,
            - ,
            *  )
```

Use Catch2 to test +=, -=, *=, /= of gsl::vectors.

8.36.2.5 TEST_CASE() [5/7]

```
TEST_CASE (
            "gsl::vector addition" ,
            "" [gsl::vector] )
```

Use Catch2 to test addition of gsl::vectors.

8.36.2.6 TEST_CASE() [6/7]

```
TEST_CASE (
            "gsl::vector and gsl::cvector addition" ,
            "" [gsl::vector][gsl::cvector] )
```

Use Catch2 to test addition of gsl::vectors and gsl::cvectors.

8.36.2.7 TEST_CASE() [7/7]

```
TEST_CASE (
            "gsl::vector and gsl::cvector scalar multiplication" ,
            "" [gsl::vector][gsl::cvector] )
```

Use Catch2 to test scalar multiplication of gsl::vectors and gsl::cvectors.