VMEC

8.52

Generated by Doxygen 1.9.1

Wed Apr 28 2021 15:29:02

Chapter 1

Educational VMEC

This is a heavily stripped-down version of the serial implementation of VMEC 8.52. It is forked from the `v251` branch of the `STELLOPT` repository.

The goal of this project is to have a version of VMEC which only computes the Stellarator MHD equilibrium and nothing more.

The `cmake` build system for stand-alone VMEC is borrowed from `hiddenSymmetries/VMEC2000` and from `ORNL-Fusion/LIBSTELL`.

1.1 Building

This is a fairly standard CMake setup, if you are used to it. Here is how it works:

- Create a directory `build` in the main folder: `mkdir build`
- Go into the `build` directory: `cd build`
- Run CMake: `cmake ..`
- Execute the actual build process: `make` (optional multi-threaded build: `make -j`)
- The VMEC executable `xvmec` is then located in `build/bin` with respect to the main folder.

1.2 Example Execution

- Change into the `test` dir: `cd test`
- Run the `Solov'ev test case`: `../build/bin/xvmec input.solovev`

1.3 External NESTOR

The free-boundary part of VMEC is the Neumann Solver for Toroidal Systems (NESTOR). Its source code is in a separate folder NESTOR. The appropriate reference is https://doi.org/10.1016/0021-9991(86)90055-0 .

This version of NESTOR can be run stand-alone. It reads its inputs from a netCDF file and writes its outputs into another netCDF file. The main executable of this stand-alone version of NESTOR is `nestor_main.f90`. The input and output files are read and written in `nestor_io.f90`.

This version of VMEC can be configured to dump the corresponding input and output files, but still run the compiled-in version of NESTOR. This is enabled via the logical flag `ldump_vacuum_ref` in `funct3d.f90`.

Also, an external NESTOR implementation can be called instead of using the compiled-in version of NESTOR. This is enabled via the logical flag `lexternal_nestor` in `funct3d.f90`. The corresponding system call to execute the external NESTOR implementation has to be specified in `nestor_executable` in `funct3d.f90`.

1.4 Angle Constraint

The poloidal angle-like coordinate is a priori not uniquely defined and needs special care. The version of VMEC from the STELLOPT repo had essentially two options for this. They were alternatively compiled in via the preprocessor flag `_HBANGLE`.

1. The Hirshman-Breslau explicit spectrally optimized Fourier series (see https://doi.org/10.1063/1.872954 for details) and
2. an unknown mixture of several constraints of the `m=1` Fourier coefficients (the logical `lconm1` is true for this constraint).

By default, the `_HBANGLE` preprocessor flag is not active and thus, the "old" `m=1` constraint is active.

This version of VMEC has most, if not all, of its preprocessor flags explicitly expanded. It became clear that it is nevertheless useful to have at least a vague idea of what parts of the code are related to the angle constraint. Therefore, those parts of VMEC related to the `m=1`constraint are marked to start with

```
! #ifndef _hbangle
```

and end with

```
! #end /* ndef _HBANGLE */
```

Chapter 2

Modules Index

2.1 Modules List

Here is a list of all documented modules with brief descriptions:

Chapter 3

Data Type Index

3.1 Data Types List

Here are the data types with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Module Documentation

5.1 line_segment Module Reference

This module containes code to create a profile constructed of line segments. These line segments are assumed to be specified such that $xx(i) < xx(i+1)$.

Functions/Subroutines

- subroutine, public **line_seg** (x, y, xx, yy, n)
- subroutine, public **line_seg_int** (x, y, xx, yy, n)
- logical function, public **line_seg_test** ()

5.1.1 Detailed Description

This module containes code to create a profile constructed of line segments. These line segments are assumed to be specified such that $xx(i) < xx(i+1)$.

5.2 mgrid_mod Module Reference

Precomputed table of magnetic field due to confimenent coils.

Functions/Subroutines

- subroutine **read_mgrid** (mgrid_file, extcur, nv, nfp, lscreen, ier_flag)
- subroutine **sum_bfield** (bfield, bf_add, cur, n1)
- subroutine **assign_bptrs** (bptr)
- subroutine **free_mgrid** (istat)

Variables

- integer, parameter **nlimset** = 2
- character(len= ∗), parameter **vn_br0** = 'br'
- character(len= ∗), parameter **vn_bp0** = 'bp'
- character(len= ∗), parameter **vn_bz0** = 'bz'
- character(len= ∗), parameter **vn_ir** = 'ir'
- character(len= ∗), parameter **vn_jz** = 'jz'
- character(len= ∗), parameter **vn_kp** = 'kp'
- character(len= ∗), parameter **vn_nfp** = 'nfp'
- character(len= ∗), parameter **vn_rmin** ='rmin'
- character(len= ∗), parameter **vn_rmax** ='rmax'
- character(len= ∗), parameter **vn_zmin** ='zmin'
- character(len= ∗), parameter **vn_zmax** ='zmax'
- character(len= ∗), parameter **vn_coilgrp** ='coil_group'
- character(len= ∗), parameter **vn_nextcur** = 'nextcur'
- character(len= ∗), parameter **vn_mgmode** ='mgrid_mode'
- character(len= ∗), parameter **vn_coilcur** = 'raw_coil_cur'
- character(len= ∗), parameter **ln_next** = 'External currents'
- integer **nr0b**
- integer **np0b**
- integer **nfper0**
- integer **nz0b**
- integer **nobd**
- integer **nobser**
- integer **nextcur**
- integer **nbfldn**
- integer **nbsets**
- integer **nbcoilsn**
- integer **nbvac**
- integer **nbcoil_max**
- integer **nlim**
- integer **nlim_max**
- integer **nsets**
- integer **nrgrid**
- integer **nzgrid**
- integer, dimension(:), allocatable **needflx**
- integer, dimension(:), allocatable **nbcoils**
- integer, dimension(:), allocatable **limitr**
- integer, dimension(:), allocatable **nsetsn**
- integer, dimension(:,:), allocatable **iconnect**
- integer, dimension(:,:), allocatable **needbfld**
- real(rprec) **rminb**
- real(rprec) **zminb**
- real(rprec) **rmaxb**
- real(rprec) **zmaxb**
- real(rprec) **delrb**
- real(rprec) **delzb**
- real(rprec) **rx1**
- real(rprec) **rx2**
- real(rprec) **zy1**
- real(rprec) **zy2**
- real(rprec) **condif**
- real(rprec), dimension(:,:), allocatable, target **bvac**
- real(rprec), dimension(:,:,:), pointer **brvac**

- real(rprec), dimension(:,:,:), pointer **bzvac**
- real(rprec), dimension(:,:,:), pointer **bpvac**
- real(rprec), dimension(:,:), allocatable **unpsiext**
- real(rprec), dimension(:,:), allocatable **plbfld**
- real(rprec), dimension(:,:), allocatable **rbcoil**
- real(rprec), dimension(:,:), allocatable **zbcoil**
- real(rprec), dimension(:,:), allocatable **abcoil**
- real(rprec), dimension(:,:), allocatable **bcoil**
- real(rprec), dimension(:,:), allocatable **rbcoilsqr**
- real(rprec), dimension(:), allocatable **raw_coil_current**
- real(rprec), dimension(:), allocatable **xobser**
- real(rprec), dimension(:), allocatable **zobser**
- real(rprec), dimension(:), allocatable **xobsqr**
- real(rprec), dimension(:), allocatable **dsiext**
- real(rprec), dimension(:), allocatable **psiext**
- real(rprec), dimension(:), allocatable **plflux**
- real(rprec), dimension(:), allocatable **b_chi**
- character(len=300) **mgrid_path**
- character(len=300) **mgrid_path_old** = " "
- character(len=30), dimension(:), allocatable **curlabel**
- character(len=15), dimension(:), allocatable **dsilabel**
- character(len=15), dimension(:), allocatable **bloopnames**
- character(len=30) **tokid**
- real(rprec), dimension(:,:,:), allocatable **dbcoil**
- real(rprec), dimension(:,:,:), allocatable **pfcspec**
- real(rprec), dimension(:,:), allocatable **rlim**
- real(rprec), dimension(:,:), allocatable **zlim**
- real(rprec), dimension(:,:), allocatable **reslim**
- real(rprec), dimension(:,:), allocatable **seplim**
- character(len=1) **mgrid_mode**

5.2.1 Detailed Description

Precomputed table of magnetic field due to confimenent coils.

5.3 nestor_io Module Reference

Input and Output for stand-alone NESTOR.

Functions/Subroutines

- subroutine **read_nestor_inputs** (vac_file)
- subroutine **write_nestor_outputs** (vac_file, lasym, ivac, ier_flag)

Variables

- character(len=255) **input_extension**
- character(len=255) **mgrid_file**
- real(dp), dimension(:), allocatable **extcur**
- real(dp), dimension(:), allocatable **raxis**
- real(dp), dimension(:), allocatable **zaxis**
- real(dp), dimension(:), allocatable **xm**
- real(dp), dimension(:), allocatable **xn**
- real(dp), dimension(:), allocatable **rmnc**
- real(dp), dimension(:), allocatable **zmns**
- real(dp), dimension(:), allocatable **rmns**
- real(dp), dimension(:), allocatable **zmnc**
- real(dp), dimension(:), allocatable **wint**
- integer **nfp**
- integer **ntor**
- integer **mpol**
- integer **ntheta**
- integer **nzeta**
- integer **nextcur**
- integer **ier_flag**
- integer **ivac**
- integer **ivacskip**
- integer **mnmax**
- integer **vacuum_calls**
- logical **lasym**
- real(dp) **ctor**
- real(dp) **rbtor**
- real(dp) **signgs**
- integer **mnpd2_nestor**
- real(dp), dimension(:), allocatable **amatsav_nestor**
- real(dp), dimension(:), allocatable **bvecsav_nestor**
- real(dp) **bsubvvac_nestor**
- character(len= ∗), dimension(1), parameter **mn1dim** = (/'mn_mode'/)
- character(len= ∗), dimension(1), parameter **mnpotdim** = (/'mn_mode_pot'/)
- character(len= ∗), dimension(1), parameter **nzntdim** = (/'nznt'/)
- character(len= ∗), dimension(1), parameter **nzetadim** = (/'nzeta'/)
- character(len= ∗), dimension(1), parameter **nextcurim** = (/'nextcur'/)
- character(len= ∗), dimension(1), parameter **bvecsavdim** =(/'mnpd2'/)
- character(len= ∗), dimension(1), parameter **amatsavdim** =(/'mnpd2_times_mnpd2'/)
- character(len= ∗), dimension(2), parameter **r2dim** = (/'mn_mode','radius '/)
- character(len= ∗), parameter **vn_vacuum_calls** = 'vacuum_calls'
- character(len= ∗), parameter **vn_ier_flag** = "ier_flag"
- character(len= ∗), parameter **vn_mgrid** = "mgrid_file"
- character(len= ∗), parameter **vn_inputext** = "input_extension"
- character(len= ∗), parameter **vn_ivacskip** = "ivacskip"
- character(len= ∗), parameter **vn_ivac** = "ivac"
- character(len= ∗), parameter **vn_nfp** = "nfp"
- character(len= ∗), parameter **vn_ntor** = "ntor"
- character(len= ∗), parameter **vn_mpol** = "mpol"
- character(len= ∗), parameter **vn_nzeta** = "nzeta"
- character(len= ∗), parameter **vn_ntheta** = "ntheta"
- character(len= ∗), parameter **vn_mnmax** = "mnmax"
- character(len= ∗), parameter **vn_pmod** = "xm"
- character(len= ∗), parameter **vn_tmod** = "xn"

- character(len= ∗), parameter **vn_rmnc** = "rmnc"
- character(len= ∗), parameter **vn_zmns** = "zmns"
- character(len= ∗), parameter **vn_rmns** = "rmns"
- character(len= ∗), parameter **vn_zmnc** = "zmnc"
- character(len= ∗), parameter **vn_rbtor** = "rbtor"
- character(len= ∗), parameter **vn_ctor** = "ctor"
- character(len= ∗), parameter **vn_lasym** = "lasym"
- character(len= ∗), parameter **vn_signgs** = "signgs"
- character(len= ∗), parameter **vn_extcur** = "extcur"
- character(len= ∗), parameter **vn_raxis_nestor** = "raxis_nestor"
- character(len= ∗), parameter **vn_zaxis_nestor** = "zaxis_nestor"
- character(len= ∗), parameter **vn_wint** = "wint"
- character(len= ∗), parameter **vn_bsqvac** = "bsqvac"
- character(len= ∗), parameter **vn_mnpd** = "mnpd"
- character(len= ∗), parameter **vn_xmpot** = "xmpot"
- character(len= ∗), parameter **vn_xnpot** = "xnpot"
- character(len= ∗), parameter **vn_potvac** = "potvac"
- character(len= ∗), parameter **vn_brv** = "brv"
- character(len= ∗), parameter **vn_bphiv** = "bphiv"
- character(len= ∗), parameter **vn_bzv** = "bzv"
- character(len= ∗), parameter **vn_bsubvvac** = "bsubvvac"
- character(len= ∗), parameter **vn_amatsav** = "amatsav"
- character(len= ∗), parameter **vn_bvecsav** = "bvecsav"
- character(len= ∗), parameter **vn_mnpd2** = "mnpd2"
- character(len= ∗), parameter **vn_r1b** = "r1b"
- character(len= ∗), parameter **vn_rub** = "rub"
- character(len= ∗), parameter **vn_rvb** = "rvb"
- character(len= ∗), parameter **vn_z1b** = "z1b"
- character(len= ∗), parameter **vn_zub** = "zub"
- character(len= ∗), parameter **vn_zvb** = "zvb"
- character(len= ∗), parameter **vn_ruu** = "ruu"
- character(len= ∗), parameter **vn_ruv** = "ruv"
- character(len= ∗), parameter **vn_rvv** = "rvv"
- character(len= ∗), parameter **vn_zuu** = "zuu"
- character(len= ∗), parameter **vn_zuv** = "zuv"
- character(len= ∗), parameter **vn_zvv** = "zvv"
- character(len= ∗), parameter **vn_guu_b** = "guu_b"
- character(len= ∗), parameter **vn_guv_b** = "guv_b"
- character(len= ∗), parameter **vn_gvv_b** = "gvv_b"
- character(len= ∗), parameter **vn_rzb2** = "rzb2"
- character(len= ∗), parameter **vn_snr** = "snr"
- character(len= ∗), parameter **vn_snv** = "snv"
- character(len= ∗), parameter **vn_snz** = "snz"
- character(len= ∗), parameter **vn_drv** = "drv"
- character(len= ∗), parameter **vn_auu** = "auu"
- character(len= ∗), parameter **vn_auv** = "auv"
- character(len= ∗), parameter **vn_avv** = "avv"
- character(len= ∗), parameter **vn_rcosuv** = "rcosuv"
- character(len= ∗), parameter **vn_rsinuv** = "rsinuv"
- character(len= ∗), parameter **vn_brad** = "brad"
- character(len= ∗), parameter **vn_bphi** = "bphi"
- character(len= ∗), parameter **vn_bz** = "bz"
- character(len= ∗), parameter **vn_bexu** = "bexu"
- character(len= ∗), parameter **vn_bexv** = "bexv"
- character(len= ∗), parameter **vn_bexn** = "bexn"

- character(len= $*$), parameter **vn_bexni** = "bexni"
- character(len= $*$), parameter **vn_grpmn** = "grpmn"
- character(len= $*$), parameter **vn_adp** = "adp"
- character(len= $*$), parameter **vn_adm** = "adm"
- character(len= $*$), parameter **vn_cma** = "cma"
- character(len= $*$), parameter **vn_sqrtc** = "sqrtc"
- character(len= $*$), parameter **vn_sqrta** = "sqrta"
- character(len= $*$), parameter **vn_delt1u** = "delt1u"
- character(len= $*$), parameter **vn_azp1u** = "azp1u"
- character(len= $*$), parameter **vn_azm1u** = "azm1u"
- character(len= $*$), parameter **vn_cma11u** = "cma11u"
- character(len= $*$), parameter **vn_r1p** = "r1p"
- character(len= $*$), parameter **vn_r1m** = "r1m"
- character(len= $*$), parameter **vn_r0p** = "r0p"
- character(len= $*$), parameter **vn_r0m** = "r0m"
- character(len= $*$), parameter **vn_ra1p** = "ra1p"
- character(len= $*$), parameter **vn_ra1m** = "ra1m"
- character(len= $*$), parameter **vn_sqad1u** = "sqad1u"
- character(len= $*$), parameter **vn_sqad2u** = "sqad2u"
- character(len= $*$), parameter **vn_all_tlp** = "all_tlp"
- character(len= $*$), parameter **vn_all_tlm** = "all_tlm"
- character(len= $*$), parameter **vn_all_slp** = "all_slp"
- character(len= $*$), parameter **vn_all_slm** = "all_slm"
- character(len= $*$), parameter **vn_m_map** = "m_map"
- character(len= $*$), parameter **vn_n_map** = "n_map"
- character(len= $*$), parameter **vn_green** = "green"
- character(len= $*$), parameter **vn_greenp** = "greenp"
- character(len= $*$), parameter **vn_tanu** = "tanu"
- character(len= $*$), parameter **vn_tanv** = "tanv"
- character(len= $*$), parameter **vn_gstore** = "gstore"
- character(len= $*$), parameter **vn_grpmn_m_map** = "grpmn_m_map"
- character(len= $*$), parameter **vn_grpmn_n_map** = "grpmn_n_map"
- character(len= $*$), parameter **vn_imirr** = "imirr"
- character(len= $*$), parameter **vn_amatrix** = "amatrix"
- character(len= $*$), parameter **vn_potu** = "potu"
- character(len= $*$), parameter **vn_potv** = "potv"
- character(len= $*$), parameter **vn_bsubu** = "bsubu"
- character(len= $*$), parameter **vn_bsubv** = "bsubv"

5.3.1 Detailed Description

Input and Output for stand-alone NESTOR.

Chapter 6

Data Type Documentation

6.1 read_wout_mod::read_wout_file Interface Reference

Public Member Functions

- subroutine **readw_and_open** (file_or_extension, ierr, iopen)

6.1.1 Detailed Description

Definition at line 236 of file read_wout_mod.f.

Chapter 7

File Documentation

7.1 src/add_fluxes.f90 File Reference

Add the magnetic fluxes to the tangential derivatives of λ to arrive at the contravariant magnetic field components B^θ and B^ζ.

Functions/Subroutines

- subroutine add_fluxes (overg, bsupu, bsupv)

    Add the magnetic fluxes to the tangential derivatives of λ to arrive at the contravariant magnetic field components B^θ and B^ζ.

7.1.1 Detailed Description

Add the magnetic fluxes to the tangential derivatives of λ to arrive at the contravariant magnetic field components B^θ and B^ζ.

7.1.2 Function/Subroutine Documentation

7.1.2.1 add_fluxes()

```
subroutine add_fluxes (
            real(rprec), dimension(nrzt), intent(in) overg,
            real(rprec), dimension(nrzt), intent(inout) bsupu,
            real(rprec), dimension(nrzt), intent(inout) bsupv )
```

Add the magnetic fluxes to the tangential derivatives of λ to arrive at the contravariant magnetic field components B^θ and B^ζ.

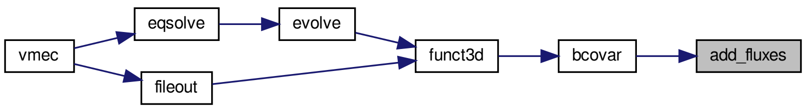
Parameters

overg	$1/\sqrt{g}$
bsupu	B^θ
bsupv	B^ζ

Definition at line 11 of file add_fluxes.f90.

Referenced by bcovar().

Here is the caller graph for this function:



7.2 src/alias.f90 File Reference

Fourier transform alias force and also return intermediate output.

Functions/Subroutines

- subroutine alias (gcons, ztemp, gcs, gsc, gcc, gss)

    Fourier transform alias force from ztemp to gcons and also return intermediate output in g(c,s)(c,s)

7.2.1 Detailed Description

Fourier transform alias force and also return intermediate output.

7.2.2 Function/Subroutine Documentation

7.2.2.1 alias()

```
subroutine alias (
            real(rprec), dimension(ns*nzeta,ntheta3), intent(out) gcons,
            real(rprec), dimension(ns*nzeta,ntheta3), intent(in) ztemp,
            real(rprec), dimension(ns,0:ntor,0:mpol1), intent(inout) gcs,
            real(rprec), dimension(ns,0:ntor,0:mpol1), intent(inout) gsc,
            real(rprec), dimension(ns,0:ntor,0:mpol1), intent(inout) gcc,
            real(rprec), dimension(ns,0:ntor,0:mpol1), intent(inout) gss )
```

Fourier transform alias force from ztemp to gcons and also return intermediate output in g(c,s)(c,s)

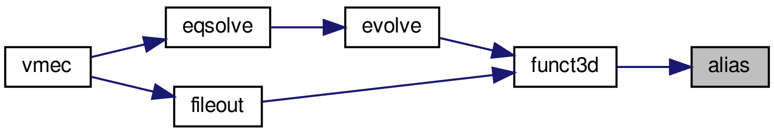
Parameters

gcons	
ztemp	
gcs	
gsc	
gcc	
gss	

Definition at line 12 of file alias.f90.

Referenced by funct3d().

Here is the caller graph for this function:



7.3 src/allocate_funct3d.f90 File Reference

allocate arrays required in funct3d()

Functions/Subroutines

- subroutine allocate_funct3d

    allocate arrays required in funct3d()

7.3.1 Detailed Description

allocate arrays required in funct3d()

7.4 src/allocate_ns.f90 File Reference

allocate arrays depending on the number of flux surfaces `ns`

Functions/Subroutines

- subroutine allocate_ns (linterp, neqs_old)

    allocate arrays depending on the number of flux surfaces `ns`

7.4.1 Detailed Description

allocate arrays depending on the number of flux surfaces `ns`

7.4.2 Function/Subroutine Documentation

7.4.2.1 allocate_ns()

```
subroutine allocate_ns (
            logical, intent(in) linterp,
            integer, intent(in) neqs_old )
```

allocate arrays depending on the number of flux surfaces `ns`

Parameters

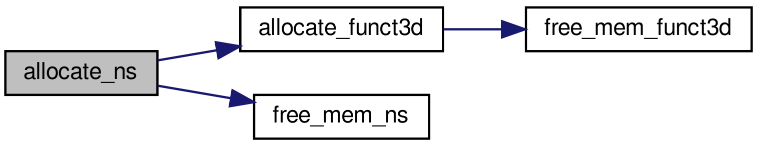
linterp	interpolate from coars to finer mesh?
neqs_old	previous number of degrees-of-freedom, i.e., Fourier coefficients for R, Z and λ

Definition at line 8 of file allocate_ns.f90.

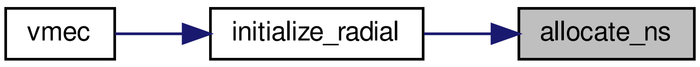
References allocate_funct3d(), and free_mem_ns().

Referenced by initialize_radial().

Here is the call graph for this function:



Here is the caller graph for this function:



7.5 src/allocate_nunv.f90 File Reference

allocate arrays depending on the number of Fourier coefficients $\mathtt{nunv}$

Functions/Subroutines

- subroutine allocate_nunv

    allocate arrays depending on the number of Fourier coefficients $\mathtt{nunv}$

7.5.1 Detailed Description

allocate arrays depending on the number of Fourier coefficients $\mathtt{nunv}$

7.6 src/aspectratio.f90 File Reference

compute aspect-ratio (independent of elongation): $A =< R > /\sqrt{< ab >}$

Functions/Subroutines

- real(rprec) function aspectratio ()

    compute aspect-ratio (independent of elongation): $A =< R > /\sqrt{< ab >}$ *where* $\pi < a >^2 = Area\ (toroidally\ averaged)$ *and* $2\pi < R > Area = Volume$

7.6.1 Detailed Description

compute aspect-ratio (independent of elongation): $A =< R > /\sqrt{< ab >}$

7.7 src/bcovar.f90 File Reference

Compute the covariant components of the magnetic field B_θ, B_ζ.

Functions/Subroutines

- subroutine bcovar (lu, lv)

    Compute the covariant components of the magnetic field B_θ*,* B_ζ*.*

7.7.1 Detailed Description

Compute the covariant components of the magnetic field B_θ, B_ζ.

7.7.2 Function/Subroutine Documentation

7.7.2.1 bcovar()

```
subroutine bcovar (
            real(rprec), dimension(nrzt,0:1), intent(inout) lu,
            real(rprec), dimension(nrzt,0:1), intent(inout) lv )
```

Compute the covariant components of the magnetic field B_θ, B_ζ.

Parameters

lu	$\partial\lambda/\partial\theta$
lv	$\partial\lambda/\partial\zeta$

R12 from RP in force

Norm, unpreconditioned R,Z forces

Norm for preconditioned R,Z forces

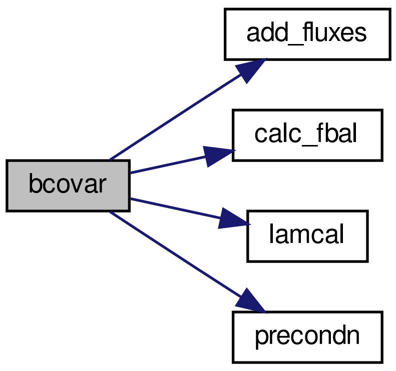
Norm for unpreconditioned Lambda force

Definition at line 8 of file bcovar.f90.

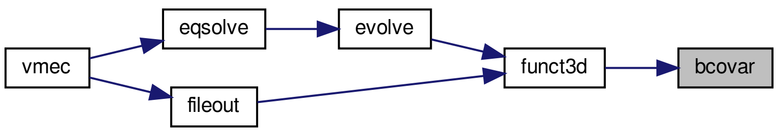
References add_fluxes(), calc_fbal(), lamcal(), and precondn().

Referenced by funct3d().

Here is the call graph for this function:



Here is the caller graph for this function:



7.8 src/bextrema.f90 File Reference

Computes minimum and maximum $|\mathbf{B}|$ along ζ between two angle lines ($\theta = 0, \pi$).

Functions/Subroutines

- subroutine bextrema (modb, bmin, bmax, nzeta, ntheta)

  Computes minimum and maximum $|\mathbf{B}|$ along ζ between two angle lines ($\theta = 0, \pi$).

7.8.1 Detailed Description

Computes minimum and maximum $|\mathbf{B}|$ along ζ between two angle lines ($\theta = 0, \pi$).

7.8.2 Function/Subroutine Documentation

7.8.2.1 bextrema()

```
subroutine bextrema (
             real(rprec), dimension(nzeta,ntheta), intent(in) modb,
             real(rprec), dimension(ntheta), intent(out) bmin,
             real(rprec), dimension(ntheta), intent(out) bmax,
             integer, intent(in) nzeta,
             integer, intent(in) ntheta )
```

Computes minimum and maximum $|\mathbf{B}|$ along ζ between two angle lines ($\theta = 0, \pi$).

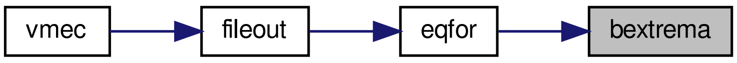
Parameters

modb	magnitude of magnetic field $|\mathbf{B}|$
bmin	minimum value of $|\mathbf{B}|$
bmax	maximum value of $|\mathbf{B}|$
nzeta	number of grid points in toroidal direction
ntheta	number of grid points in poloidal direction

Definition at line 11 of file bextrema.f90.

Referenced by eqfor().

Here is the caller graph for this function:



7.9 src/bss.f90 File Reference

Computes br, bphi, bz, bsubs on half-radial mesh.

Functions/Subroutines

- subroutine bss (r12, rs, zs, ru12, zu12, bsubs, bsupu, bsupv, br, bphi, bz)

  Computes br, bphi, bz, bsubs on half-radial mesh.

7.9.1 Detailed Description

Computes br, bphi, bz, bsubs on half-radial mesh.

7.9.2 Function/Subroutine Documentation

7.9.2.1 bss()

```
subroutine bss (
             real(rprec), dimension(nrzt), intent(in) r12,
             real(rprec), dimension(nrzt), intent(in) rs,
             real(rprec), dimension(nrzt), intent(in) zs,
             real(rprec), dimension(nrzt), intent(in) ru12,
             real(rprec), dimension(nrzt), intent(in) zu12,
             real(rprec), dimension(nrzt), intent(out) bsubs,
             real(rprec), dimension(nrzt), intent(in) bsupu,
             real(rprec), dimension(nrzt), intent(in) bsupv,
             real(rprec), dimension(nrzt), intent(out) br,
             real(rprec), dimension(nrzt), intent(out) bphi,
             real(rprec), dimension(nrzt), intent(out) bz )
```

Computes br, bphi, bz, bsubs on half-radial mesh.

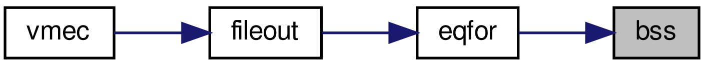
Parameters

r12	R^2
rs	$\partial R/\partial s$
zs	$\partial Z/\partial s$
ru12	$(\partial R/\partial \theta)^2$
zu12	$(\partial Z/\partial \theta)^2$
bsubs	covariant component of magnetic field B_s
bsupu	contravariant component of magnetic field B^θ
bsupv	contravariant component of magnetic field B^ζ
br	cylindrical component of magnetic field B^R
bphi	cylindrical component of magnetic field B^φ
bz	cylindrical component of magnetic field B^Z

Definition at line 17 of file bss.f90.

Referenced by eqfor().

Here is the caller graph for this function:



7.10 src/calc_fbal.f90 File Reference

Compute flux-surface averaged radial force balance $\nabla p - <\mathbf{j}\times\mathbf{B}>$.

Functions/Subroutines

- subroutine calc_fbal (bsubu, bsubv)

  Compute flux-surface averaged radial force balance $\nabla p - <\mathbf{j}\times\mathbf{B}>$.

7.10.1 Detailed Description

Compute flux-surface averaged radial force balance $\nabla p - <\mathbf{j}\times\mathbf{B}>$.

7.10.2 Function/Subroutine Documentation

7.10.2.1 calc_fbal()

```
subroutine calc_fbal (
            real(dp), dimension(1:nrzt), intent(in) bsubu,
            real(dp), dimension(1:nrzt), intent(in) bsubv )
```

Compute flux-surface averaged radial force balance $\nabla p - < \mathbf{j} \times \mathbf{B} >$.

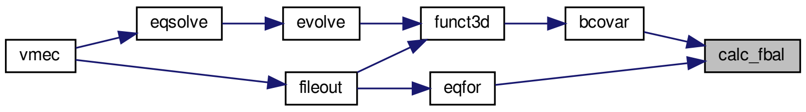
Parameters

bsubu	covariant component of magnetic field B_θ
bsubv	covariant component of magnetic field B_ζ

Definition at line 8 of file calc_fbal.f90.

Referenced by bcovar(), and eqfor().

Here is the caller graph for this function:



7.11 src/convert.f90 File Reference

Convert internal mode representation to standard form for output (coefficients of cos(mu-nv), sin(mu-nv) without internal `mscale` , `nscale` norms).

Functions/Subroutines

- subroutine convert (rmnc, zmns, lmns, rmns, zmnc, lmnc, rzl_array, js)

  Convert internal mode representation to standard form for output (coefficients of cos(mu-nv), sin(mu-nv) without internal `mscale` , `nscale` norms).

7.11.1 Detailed Description

Convert internal mode representation to standard form for output (coefficients of cos(mu-nv), sin(mu-nv) without internal `mscale` , `nscale` norms).

7.11.2 Function/Subroutine Documentation

7.11.2.1 convert()

```
subroutine convert (
            real(rprec), dimension(mnmax), intent(out) rmnc,
            real(rprec), dimension(mnmax), intent(out) zmns,
            real(rprec), dimension(mnmax), intent(out) lmns,
            real(rprec), dimension(mnmax), intent(out) rmns,
            real(rprec), dimension(mnmax), intent(out) zmnc,
            real(rprec), dimension(mnmax), intent(out) lmnc,
            real(rprec), dimension(ns, 0:ntor, 0:mpol1, 3*ntmax), intent(in) rzl_array,
            integer, intent(in) js )
```

Convert internal mode representation to standard form for output (coefficients of cos(mu-nv), sin(mu-nv) without internal `mscale`, `nscale` norms).

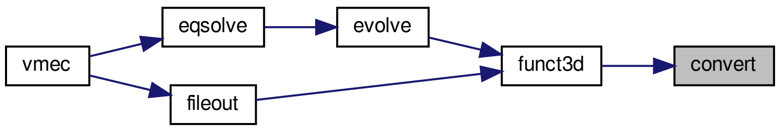
Parameters

rmnc	stellarator-symmetric Fourier coefficients of R
zmns	stellarator-symmetric Fourier coefficients of Z
lmns	stellarator-symmetric Fourier coefficients of λ
rmns	non-stellarator-symmetric Fourier coefficients of R
zmnc	non-stellarator-symmetric Fourier coefficients of Z
lmnc	non-stellarator-symmetric Fourier coefficients of λ
rzl_array	state vector (all Fourier coefficients) of VMEC
js	index of flux surface at which to do the conversion

Definition at line 16 of file convert.f90.

Referenced by funct3d().

Here is the caller graph for this function:



7.12 src/data/fbal.f90 File Reference

Variables

- real(dp), dimension(:), allocatable **fbal::rzu_fac**
- real(dp), dimension(:), allocatable **fbal::rru_fac**
- real(dp), dimension(:), allocatable **fbal::frcc_fac**
- real(dp), dimension(:), allocatable **fbal::fzsc_fac**

7.13 src/data/realspace.f90 File Reference

Variables

- real(rprec), dimension(:,:), allocatable **realspace::r1**
- real(rprec), dimension(:,:), allocatable **realspace::ru**
- real(rprec), dimension(:,:), allocatable **realspace::rv**
- real(rprec), dimension(:,:), allocatable, target **realspace::z1**
- real(rprec), dimension(:,:), allocatable **realspace::zu**
- real(rprec), dimension(:,:), allocatable **realspace::zv**
- real(rprec), dimension(:,:), allocatable **realspace::rcon**
- real(rprec), dimension(:,:), allocatable **realspace::zcon**
- real(rprec), dimension(:), allocatable **realspace::guu**
- real(rprec), dimension(:), allocatable **realspace::guv**
- real(rprec), dimension(:), allocatable **realspace::gvv**
- real(rprec), dimension(:), allocatable **realspace::ru0**
- real(rprec), dimension(:), allocatable **realspace::zu0**
- real(rprec), dimension(:), allocatable **realspace::gcon**
- real(rprec), dimension(:), allocatable **realspace::rcon0**
- real(rprec), dimension(:), allocatable **realspace::zcon0**
- real(rprec), dimension(:), allocatable realspace::phip

  radial derivative of phi/(2∗pi) on half-grid
- real(rprec), dimension(:), allocatable realspace::chip

  radial derivative of chi/(2∗pi) on half-grid
- real(rprec), dimension(:), allocatable realspace::shalf

  sqrt(s) ,two-dimensional array on half-grid
- real(rprec), dimension(:), allocatable realspace::sqrts

  sqrt(s), two-dimensional array on full-grid
- real(rprec), dimension(:), allocatable realspace::wint

  two-dimensional array for normalizing angle integrations
- real(rprec), dimension(:,:), allocatable, target **realspace::extra1**
- real(rprec), dimension(:,:), allocatable, target **realspace::extra2**
- real(rprec), dimension(:,:), allocatable, target **realspace::extra3**
- real(rprec), dimension(:,:), allocatable, target **realspace::extra4**

7.14 src/data/stel_constants.f File Reference

Variables

- real(dp), parameter **stel_constants::pi** =3.14159265358979323846264338328_dp
- real(dp), parameter **stel_constants::pio2** =pi/2
- real(dp), parameter **stel_constants::twopi** =2∗pi
- real(dp), parameter **stel_constants::sqrt2** =1.41421356237309504880168872_dp
- real(dp), parameter **stel_constants::degree** =twopi / 360
- real(dp), parameter **stel_constants::one** =1
- real(dp), parameter **stel_constants::zero** =0
- real(dp), parameter **stel_constants::mu0** = 2 ∗ twopi ∗ 1.0e-7_dp

7.15 src/data/stel_kinds.f File Reference

Variables

- integer, parameter **stel_kinds::rprec** = SELECTED_REAL_KIND(12, 100)
- integer, parameter **stel_kinds::iprec** = SELECTED_INT_KIND(8)
- integer, parameter **stel_kinds::cprec** = KIND((1.0_rprec, 1.0_rprec))
- integer, parameter **stel_kinds::dp** = rprec

7.16 src/data/vforces.f90 File Reference

Variables

- real(rprec), dimension(:), allocatable, target **vforces::armn**
- real(rprec), dimension(:), allocatable, target **vforces::azmn**
- real(rprec), dimension(:), allocatable, target **vforces::brmn**
- real(rprec), dimension(:), allocatable, target **vforces::bzmn**
- real(rprec), dimension(:), allocatable, target **vforces::blmn**
- real(rprec), dimension(:), allocatable, target **vforces::crmn**
- real(rprec), dimension(:), allocatable, target **vforces::czmn**
- real(rprec), dimension(:), allocatable, target **vforces::clmn**
- real(rprec), dimension(:), pointer **vforces::armn_e**
- real(rprec), dimension(:), pointer **vforces::armn_o**
- real(rprec), dimension(:), pointer **vforces::azmn_e**
- real(rprec), dimension(:), pointer **vforces::azmn_o**
- real(rprec), dimension(:), pointer **vforces::brmn_e**
- real(rprec), dimension(:), pointer **vforces::brmn_o**
- real(rprec), dimension(:), pointer **vforces::bzmn_e**
- real(rprec), dimension(:), pointer **vforces::bzmn_o**
- real(rprec), dimension(:), pointer **vforces::blmn_e**
- real(rprec), dimension(:), pointer **vforces::blmn_o**
- real(rprec), dimension(:), pointer **vforces::crmn_e**
- real(rprec), dimension(:), pointer **vforces::crmn_o**
- real(rprec), dimension(:), pointer **vforces::czmn_e**
- real(rprec), dimension(:), pointer **vforces::czmn_o**
- real(rprec), dimension(:), pointer **vforces::clmn_e**
- real(rprec), dimension(:), pointer **vforces::clmn_o**

7.17 src/data/vmec_dim.f90 File Reference

Variables

- integer **vmec_dim::mpol1**
- integer **vmec_dim::ntor1**
- integer **vmec_dim::mnmax**
- integer **vmec_dim::ntheta1**
- integer **vmec_dim::ntheta2**
- integer **vmec_dim::ntheta3**
- integer **vmec_dim::nznt**
- integer **vmec_dim::nrzt**
- integer **vmec_dim::mns**
- integer **vmec_dim::mnsize**
- integer **vmec_dim::mnmax_nyq**
- integer **vmec_dim::ns**
- integer **vmec_dim::ns1**
- integer **vmec_dim::ns_maxval**

7.18 src/data/vmec_input.f90 File Reference

Functions/Subroutines

- subroutine **vmec_input::read_indata_namelist** (iunit, istat)
- subroutine **vmec_input::write_indata_namelist** (iunit, istat)

Variables

- integer, parameter **vmec_input::mpol_default** = 6
- integer, parameter **vmec_input::ntor_default** = 0
- integer, parameter **vmec_input::ns_default** = 31
- integer, parameter **vmec_input::niter_default** = 100
- real(rprec), parameter **vmec_input::ftol_default** = 1.E-10_dp
- integer **vmec_input::nfp**
- integer **vmec_input::ncurr**
- integer **vmec_input::nstep**
- integer **vmec_input::nvacskip**
- integer **vmec_input::mpol**
- integer **vmec_input::ntor**
- integer **vmec_input::ntheta**
- integer **vmec_input::nzeta**
- integer **vmec_input::mfilter_fbdy**
- integer **vmec_input::nfilter_fbdy**
- integer, dimension(100) **vmec_input::ns_array**
- integer, dimension(100) **vmec_input::niter_array**
- real(rprec), dimension(100) **vmec_input::ftol_array**
- real(rprec), dimension(-ntord:ntord, 0:mpol1d) **vmec_input::rbc**
- real(rprec), dimension(-ntord:ntord, 0:mpol1d) **vmec_input::zbs**
- real(rprec), dimension(-ntord:ntord, 0:mpol1d) **vmec_input::rbs**
- real(rprec), dimension(-ntord:ntord, 0:mpol1d) **vmec_input::zbc**
- real(rprec) **vmec_input::curtor**
- real(rprec) **vmec_input::delt**
- real(rprec) **vmec_input::tcon0**
- real(rprec) **vmec_input::gamma**
- real(rprec) **vmec_input::bloat**
- real(rprec) **vmec_input::pres_scale**
- real(rprec) vmec_input::spres_ped

  value of s beyond which pressure profile is flat (pedestal)
- real(rprec) vmec_input::phiedge

  value of real toroidal flux at plasma edge (s=1)
- real(rprec), dimension(0:20) vmec_input::am

  array of coefficients in phi-series for mass (NWT/m∗∗2)
- real(rprec), dimension(0:20) vmec_input::ai

  array of coefficients in phi-series for iota (ncurr=0)
- real(rprec), dimension(0:20) vmec_input::ac

  array of coefficients in phi-series for the quantity d(Icurv)/ds = toroidal current density ∗ Vprime, so Icurv(s) = Itor(s) (used for ncurr=1)
- real(rprec), dimension(1:20) **vmec_input::aphi**
- character(len=20) **vmec_input::pcurr_type**
- character(len=20) **vmec_input::piota_type**
- character(len=20) **vmec_input::pmass_type**

- real(rprec), dimension(ndatafmax) **vmec_input::am_aux_s**
- real(rprec), dimension(ndatafmax) **vmec_input::am_aux_f**
- real(rprec), dimension(ndatafmax) **vmec_input::ai_aux_s**
- real(rprec), dimension(ndatafmax) **vmec_input::ai_aux_f**
- real(rprec), dimension(ndatafmax) **vmec_input::ac_aux_s**
- real(rprec), dimension(ndatafmax) **vmec_input::ac_aux_f**
- real(rprec), dimension(0:ntord) **vmec_input::raxis_cc**
- real(rprec), dimension(0:ntord) **vmec_input::raxis_cs**
- real(rprec), dimension(0:ntord) **vmec_input::zaxis_cc**
- real(rprec), dimension(0:ntord) **vmec_input::zaxis_cs**
- real(rprec), dimension(nigroup) **vmec_input::extcur**
- logical **vmec_input::lfreeb**
- logical **vmec_input::lasym**
- logical **vmec_input::lbsubs**
- character(len=200) **vmec_input::mgrid_file**
- character(len=100) **vmec_input::input_extension**

7.19 src/data/vmec_io.f90 File Reference

Variables

- real(rprec) **vmec_io::volavgb**
- real(rprec) **vmec_io::ionlarmor**
- real(rprec) **vmec_io::aminor_p**
- real(rprec) **vmec_io::rmajor_p**
- real(rprec) **vmec_io::betatot**
- real(rprec) **vmec_io::betapol**
- real(rprec) **vmec_io::betator**
- real(rprec) **vmec_io::betaxis**
- real(rprec) **vmec_io::b0**
- real(rprec) **vmec_io::volume_p**
- real(rprec) **vmec_io::cross_area_p**
- real(rprec) **vmec_io::surf_area_p**
- real(rprec) **vmec_io::circum_p**
- real(rprec) **vmec_io::kappa_p**
- real(rprec) **vmec_io::rmax_surf**
- real(rprec) **vmec_io::rmin_surf**
- real(rprec) **vmec_io::zmax_surf**

7.20 src/data/vmec_main.f90 File Reference

Variables

- real(rprec), dimension(:,:), allocatable **vmec_main::ard**
- real(rprec), dimension(:,:), allocatable **vmec_main::arm**
- real(rprec), dimension(:,:), allocatable **vmec_main::brd**
- real(rprec), dimension(:,:), allocatable **vmec_main::brm**
- real(rprec), dimension(:,:), allocatable **vmec_main::azd**
- real(rprec), dimension(:,:), allocatable **vmec_main::azm**
- real(rprec), dimension(:,:), allocatable **vmec_main::bzd**

- real(rprec), dimension(:,:), allocatable **vmec_main::bzm**
- real(rprec), dimension(:,:), allocatable **vmec_main::bmin**
- real(rprec), dimension(:,:), allocatable **vmec_main::bmax**
- real(rprec), dimension(:), allocatable **vmec_main::crd**
- real(rprec), dimension(:), allocatable **vmec_main::iotaf**
- real(rprec), dimension(:), allocatable **vmec_main::phipf**
- real(rprec), dimension(:), allocatable **vmec_main::chipf**
- real(rprec), dimension(:), allocatable **vmec_main::phi**
- real(rprec), dimension(:), allocatable **vmec_main::beta_vol**
- real(rprec), dimension(:), allocatable **vmec_main::jcuru**
- real(rprec), dimension(:), allocatable **vmec_main::jcurv**
- real(rprec), dimension(:), allocatable **vmec_main::jdotb**
- real(rprec), dimension(:), allocatable **vmec_main::buco**
- real(rprec), dimension(:), allocatable **vmec_main::bvco**
- real(rprec), dimension(:), allocatable **vmec_main::bdotgradv**
- real(rprec), dimension(:), allocatable **vmec_main::equif**
- real(rprec), dimension(:), allocatable **vmec_main::specw**
- real(rprec), dimension(:), allocatable **vmec_main::tcon**
- real(rprec), dimension(:), allocatable **vmec_main::psi**
- real(rprec), dimension(:), allocatable **vmec_main::yellip**
- real(rprec), dimension(:), allocatable **vmec_main::yinden**
- real(rprec), dimension(:), allocatable **vmec_main::ytrian**
- real(rprec), dimension(:), allocatable **vmec_main::yshift**
- real(rprec), dimension(:), allocatable **vmec_main::ygeo**
- real(rprec), dimension(:), allocatable **vmec_main::overr**
- real(rprec), dimension(:), allocatable **vmec_main::sm**
- real(rprec), dimension(:), allocatable **vmec_main::sp**
- real(rprec), dimension(:), allocatable **vmec_main::pres**
- real(rprec), dimension(:), allocatable **vmec_main::vp**
- real(rprec), dimension(:), allocatable **vmec_main::jpar2**
- real(rprec), dimension(:), allocatable **vmec_main::jperp2**
- real(rprec), dimension(:), allocatable **vmec_main::bdotb**
- real(rprec), dimension(:), allocatable **vmec_main::blam**
- real(rprec), dimension(:), allocatable **vmec_main::clam**
- real(rprec), dimension(:), allocatable **vmec_main::dlam**
- real(rprec), dimension(:), allocatable **vmec_main::vpphi**
- real(rprec), dimension(:), allocatable **vmec_main::presgrad**
- real(rprec), dimension(:), allocatable **vmec_main::bdamp**
- real(rprec), dimension(:), allocatable **vmec_main::bucof**
- real(rprec), dimension(:), allocatable **vmec_main::bvcof**
- real(rprec), dimension(:), allocatable **vmec_main::chi**
- real(rprec), dimension(:), allocatable vmec_main::presf

  pressure profile on full-grid, mass/phip∗∗gamma
- real(rprec), dimension(:), allocatable vmec_main::chips

  poloidal flux (same as chip), one-dimensional array
- real(rprec), dimension(:), allocatable vmec_main::phips

  toroidal flux (same as phip), one-dimensional array
- real(rprec), dimension(:), allocatable vmec_main::iotas

  rotational transform , on half radial mesh
- real(rprec), dimension(:), allocatable vmec_main::icurv

  (-)toroidal current inside flux surface (vanishes like s)
- real(rprec), dimension(:), allocatable vmec_main::mass

  mass profile on half-grid
- real(rprec), dimension(:,:,:,:), allocatable **vmec_main::faclam**

- real(rprec), dimension(:,:,:,:), allocatable **vmec_main::faclam0**
- real(rprec), dimension(:,:), allocatable **vmec_main::bsqsav**
- real(rprec), dimension(:), allocatable **vmec_main::bredge**
- real(rprec), dimension(:), allocatable **vmec_main::bpedge**
- real(rprec), dimension(:), allocatable **vmec_main::bzedge**
- real(rprec), dimension(:), allocatable **vmec_main::xcl0**
- real(rprec), dimension(0:mpol1d, 3) **vmec_main::xmpq**
- real(rprec), dimension(0:mpol1d) **vmec_main::faccon**
- real(rprec) vmec_main::hs

  radial mesh size increment
- real(rprec) **vmec_main::currv**
- real(rprec) **vmec_main::aspect**
- real(rprec) **vmec_main::ohs**
- real(rprec) **vmec_main::voli**
- real(rprec) **vmec_main::r00**
- real(rprec) **vmec_main::r0scale**
- real(rprec) **vmec_main::z00**
- real(rprec) **vmec_main::fsqsum0**
- real(rprec) **vmec_main::fnorm**
- real(rprec) **vmec_main::fsqr** =1
- real(rprec) **vmec_main::fsqz** =1
- real(rprec) **vmec_main::fsql** =1
- real(rprec) **vmec_main::fnorm1**
- real(rprec) **vmec_main::fnorml**
- real(rprec) **vmec_main::fsqr1**
- real(rprec) **vmec_main::fsqz1**
- real(rprec) **vmec_main::fsql1**
- real(rprec) **vmec_main::fsq**
- real(rprec) **vmec_main::fedge**
- real(rprec) **vmec_main::wb**
- real(rprec) **vmec_main::wp**
- real(rprec) **vmec_main::router**
- real(rprec) **vmec_main::rinner**
- real(rprec) **vmec_main::ftolv**
- real(rprec) vmec_main::otav

  time-step algorithm
- real(rprec), dimension(ndamp) **vmec_main::otau**
- real(rprec), dimension(:,:,:), allocatable, target **vmec_main::rmn_bdy**
- real(rprec), dimension(:,:,:), allocatable, target **vmec_main::zmn_bdy**
- real(rprec), dimension(:), allocatable **vmec_main::bsubu0**
- real(rprec), dimension(:), allocatable **vmec_main::dbsq**
- real(rprec), dimension(:), allocatable **vmec_main::rbsq**
- real(rprec) **vmec_main::rbtor**
- real(rprec) **vmec_main::rbtor0**
- real(rprec) **vmec_main::ctor**
- real(rprec) **vmec_main::delbsq**
- real(rprec) **vmec_main::res0**
- real(rprec) **vmec_main::delt0r**
- real(rprec), dimension(ndatafmax) **vmec_main::spfa**
- real(rprec), dimension(ndatafmax) **vmec_main::spfa2**
- real(rprec), dimension(ndatafmax) **vmec_main::hp**
- real(rprec), dimension(ndatafmax) **vmec_main::sifa**
- real(rprec), dimension(ndatafmax) **vmec_main::sifa2**
- real(rprec), dimension(ndatafmax) **vmec_main::hi**

- logical **vmec_main::lthreed**
- logical **vmec_main::lconm1**
- logical vmec_main::lflip

  from init_geometry
- integer, dimension(:), allocatable vmec_main::ireflect

  two-dimensional array for computing 2pi-v angle
- integer **vmec_main::multi_ns_grid**
- integer **vmec_main::itfsq**
- integer **vmec_main::ndatap**
- integer **vmec_main::ndatai**
- integer vmec_main::niterv

  max iterations for current multi-grid iteration
- integer vmec_main::neqs

  total number of equations to evolve (size of xc)
- integer vmec_main::irzloff

  offset in xc array between R,Z,L components
- integer vmec_main::iequi

  counter used to call -EQFOR- at end of run
- integer vmec_main::ijacob

  counter for number of times jacobian changes sign
- integer vmec_main::irst

  "counter" monitoring sign of jacobian; resets R, Z, and Lambda when jacobian changes sign and decreases time step
- integer vmec_main::iter1

  number of iterations at which the currently active evolution was branched off from
- integer vmec_main::iter2

  total number of iterations
- integer vmec_main::ivac

  counts number of free-boundary iterations
- integer **vmec_main::vacuum_calls** = 0

7.21 src/data/vmec_params.f90 File Reference

Variables

- integer, parameter vmec_params::meven = 0

  parity selection label for even poloidal modes of R and Z
- integer, parameter vmec_params::modd = 1

  parity selection label for odd poloidal modes of R and Z
- integer, parameter vmec_params::ndamp = 10

  number of iterations over which damping is averaged
- integer, parameter **vmec_params::ns4** = 25
- integer, dimension(0:mpold), parameter vmec_params::jmin1 = (/ 1,1,(2,ink=2,mpold) /)

  starting js(m) values where R,Z are non-zero
- integer, dimension(0:mpold), parameter vmec_params::jmin2 = (/ 1,2,(2,ink=2,mpold) /)

  starting js(m) values for which R,Z are evolved
- integer, dimension(0:mpold), parameter vmec_params::jlam = (/ 2,2,(2,ink=2,mpold) /)

  starting js(m) values for which Lambda is evolved
- integer, parameter **vmec_params::norm_term_flag** = 0
- integer, parameter **vmec_params::bad_jacobian_flag** = 1
- integer, parameter **vmec_params::jac75_flag** = 4

- integer, parameter **vmec_params::input_error_flag** = 5
- integer, parameter **vmec_params::phiedge_error_flag** = 7
- integer, parameter **vmec_params::ns_error_flag** = 8
- integer, parameter **vmec_params::misc_error_flag** = 9
- integer, parameter **vmec_params::successful_term_flag** = 11
- integer, parameter **vmec_params::restart_flag** = 1
- integer, parameter **vmec_params::readin_flag** = 2
- integer, parameter **vmec_params::timestep_flag** = 4
- integer, parameter **vmec_params::output_flag** = 8
- integer, parameter **vmec_params::cleanup_flag** = 16
- integer, parameter **vmec_params::reset_jacdt_flag** = 32
- real(rprec), parameter **vmec_params::pdamp** = 0.05_dp
- character(len= ∗), parameter **vmec_params::version_** = '8.52'
- integer vmec_params::ntmax

  number of contributing Fourier basis function (can be 1, 2 or 4); assigned in read_indata()
- integer **vmec_params::rcc**
- integer **vmec_params::rss**
- integer **vmec_params::rsc**
- integer **vmec_params::rcs**
- integer **vmec_params::zsc**
- integer **vmec_params::zcs**
- integer **vmec_params::zcc**
- integer **vmec_params::zss**
- integer **vmec_params::mnyq**
- integer **vmec_params::nnyq**
- integer, dimension(:), allocatable **vmec_params::uminus**
- real(rprec), dimension(:), allocatable vmec_params::mscale

  array for norming theta-trig functions (internal use only) so that the discrete SUM[cos(mu)∗cos(m'u)] = .5 delta(m,m')
- real(rprec), dimension(:), allocatable vmec_params::nscale

  array for norming zeta -trig functions (internal use only)
- real(rprec) vmec_params::signgs

  sign of Jacobian : must be =1 (right-handed) or =-1 (left-handed)
- real(rprec) **vmec_params::lamscale** =1
- integer, parameter vmec_params::m0 =0

  from totzsp
- integer, parameter vmec_params::m1 =1

  from totzsp
- integer, parameter vmec_params::n0 =0

  from totzsp

7.22 src/data/vmec_persistent.f90 File Reference

Variables

- integer, dimension(:), allocatable **vmec_persistent::ixm**
- integer, dimension(:), allocatable **vmec_persistent::jmin3**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosmu**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::sinmu**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosmum**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::sinmum**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosmumi**

- real(rprec), dimension(:,:), allocatable **vmec_persistent::sinmumi**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosnv**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::sinnv**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosnvn**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::sinnvn**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosmui**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::sinmui**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosmui3**
- real(rprec), dimension(:,:), allocatable **vmec_persistent::cosmumi3**
- real(rprec), dimension(:), allocatable, target **vmec_persistent::xm**
- real(rprec), dimension(:), allocatable, target **vmec_persistent::xn**
- real(rprec), dimension(:), allocatable, target **vmec_persistent::xm_nyq**
- real(rprec), dimension(:), allocatable, target **vmec_persistent::xn_nyq**
- real(rprec), dimension(:), allocatable **vmec_persistent::cos01**
- real(rprec), dimension(:), allocatable **vmec_persistent::sin01**

7.23 src/data/vmercier.f90 File Reference

Variables

- real(rprec), dimension(nsd) **vmercier::dshear**
- real(rprec), dimension(nsd) **vmercier::dwell**
- real(rprec), dimension(nsd) **vmercier::dcurr**
- real(rprec), dimension(nsd) **vmercier::dmerc**
- real(rprec), dimension(nsd) **vmercier::dgeod**

7.24 src/data/vparams.f90 File Reference

Variables

- integer, parameter vparams::nsd = 10001

  maximum number of radial nodes
- integer, parameter vparams::mpold = 101

  maximum number of poloidal harmonics (in r,z,lam fourier series)
- integer, parameter vparams::ntord = 101

  maximum number of toroidal harmonics
- integer, parameter **vparams::ndatafmax** = 101
- integer, parameter **vparams::nstore_seq** = 100
- integer, parameter **vparams::mpol1d** = mpold - 1
- integer, parameter **vparams::ntor1d** = ntord + 1
- integer, parameter **vparams::nthreed0** = 9
- integer, parameter **vparams::indata0** = nthreed0 + 2
- integer, parameter **vparams::nwout0** = nthreed0 + 3
- integer, parameter **vparams::jxbout0** = nthreed0 + 4
- integer, parameter **vparams::nfort18** = 18
- integer, parameter **vparams::nmercier0** = 52
- integer **vparams::nthread**
- real(rprec), parameter **vparams::c1pm2** = 1.e-2_dp
- real(rprec), parameter **vparams::cp15** = 0.15_dp
- real(rprec), parameter **vparams::cp25** = 0.25_dp

- real(rprec), parameter **vparams::cp5** = 0.50_dp
- real(rprec), parameter **vparams::c1pm8** = 1.0e-8_dp
- real(rprec), parameter **vparams::cbig** = 0.9e30_dp
- real(rprec), parameter **vparams::c2p0** = 2
- real(rprec), parameter **vparams::c3p0** = 3
- real(rprec), parameter **vparams::cp05** = 0.05_dp
- real(rprec), parameter **vparams::c1pm13** = 1.0e-13_dp
- real(rprec), parameter **vparams::osqrt2** = 0.707106781186547462_dp

7.25 src/data/vsvd0.f90 File Reference

Variables

- integer, parameter vsvd0::nigroup = 100

  number of external current groups

7.26 src/data/xstuff.f90 File Reference

Variables

- real(rprec), dimension(:), allocatable xstuff::gc

  stacked array of R, Z, Lambda Spectral force coefficients (see above for stack order)
- real(rprec), dimension(:), allocatable, target xstuff::xc

  stacked array of scaled R, Z, Lambda Fourier coefficients (see above for stack order)
- real(rprec), dimension(:), allocatable xstuff::xcdot

  "velocity": change of Fourier coefficients per time step
- real(rprec), dimension(:), allocatable **xstuff::xsave**
- real(rprec), dimension(:), allocatable xstuff::xstore

  backup copy of last-known-good xc
- real(rprec), dimension(:), allocatable **xstuff::scalxc**

7.27 src/elongation.f90 File Reference

Compute Waist thickness and height in $\varphi = 0, \pi$ symmetry planes.

Functions/Subroutines

- subroutine elongation (r1, z1, waist, height)

  Compute Waist thickness and height in $\varphi = 0, \pi$ symmetry planes.

7.27.1 Detailed Description

Compute Waist thickness and height in $\varphi = 0, \pi$ symmetry planes.

7.27.2 Function/Subroutine Documentation

7.27.2.1 elongation()

```
subroutine elongation (
            real(rprec), dimension(ns,nzeta,ntheta3,0:1), intent(in) r1,
            real(rprec), dimension(ns,nzeta,ntheta3,0:1), intent(in) z1,
            real(rprec), dimension(2), intent(out) waist,
            real(rprec), dimension(2), intent(out) height )
```

Compute Waist thickness and height in $\varphi = 0, \pi$ symmetry planes.

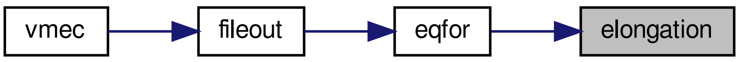
Parameters

r1	R
z1	Z
waist	
height	

Definition at line 10 of file elongation.f90.

Referenced by eqfor().

Here is the caller graph for this function:



7.28 src/eqfor.f90 File Reference

Basis physics analysis and evaluaton of force balance. This is where most of the contents of the `threed1` output file is computed.

Functions/Subroutines

- subroutine eqfor (br, bz, bsubu, bsubv, tau, rzl_array, ier_flag)

  Basis physics analysis and evaluaton of force balance. This is where most of the contents of the `threed1` output file is computed.

7.28.1 Detailed Description

Basis physics analysis and evaluaton of force balance. This is where most of the contents of the `threed1` output file is computed.

7.28.2 Function/Subroutine Documentation

7.28.2.1 eqfor()

```
subroutine eqfor (
            real(rprec), dimension(nrzt), intent(out) br,
            real(rprec), dimension(nrzt), intent(out) bz,
            real(rprec), dimension(ns,nznt,0:1), intent(in) bsubu,
            real(rprec), dimension(ns,nznt,0:1), intent(in) bsubv,
            real(rprec), dimension(nrzt), intent(out) tau,
            real(rprec), dimension(ns,0:ntor,0:mpol1,3*ntmax), intent(in), target rzl_array,
            integer ier_flag )
```

Basis physics analysis and evaluaton of force balance. This is where most of the contents of the `threed1` output file is computed.

Parameters

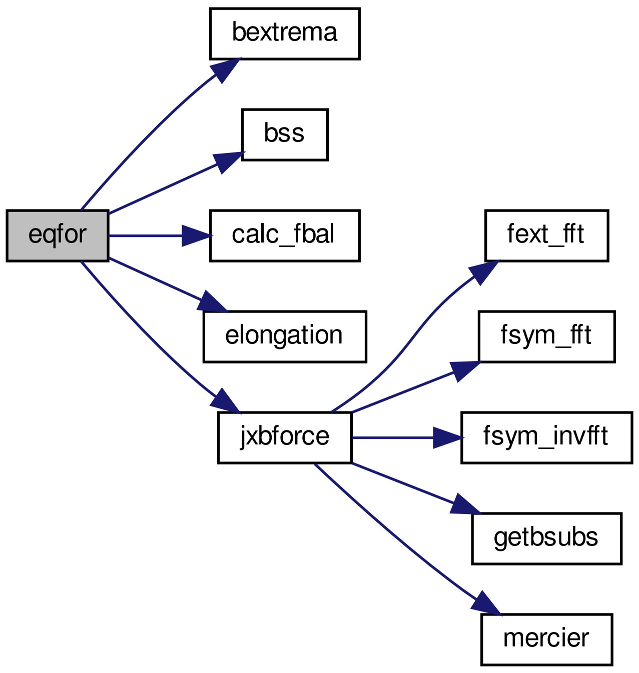
br	cylindrical component of magnetic field B^R
bz	cylindrical component of magnetic field B^Z
bsubu	covariant component of magnetic field B_θ
bsubv	covariant component of magnetic field B_ζ
tau	Jacobian $\sqrt{g} = R\tau$
rzl_array	state vector (all Fourier coefficients) of VMEC
ier_flag	error flag

Definition at line 15 of file eqfor.f90.

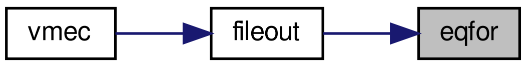
References bextrema(), bss(), calc_fbal(), elongation(), and jxbforce().

Referenced by fileout().

Here is the call graph for this function:



Here is the caller graph for this function:



7.29 src/eqsolve.f90 File Reference

Iteratively evolve the Fourier coefficients that specify the equilibrium.

Functions/Subroutines

- subroutine eqsolve (ier_flag)

  Iteratively evolve the Fourier coefficients that specify the equilibrium.

7.29.1 Detailed Description

Iteratively evolve the Fourier coefficients that specify the equilibrium.

7.29.2 Function/Subroutine Documentation

7.29.2.1 eqsolve()

```
subroutine eqsolve (
             integer, intent(inout) ier_flag )
```

Iteratively evolve the Fourier coefficients that specify the equilibrium.

Parameters

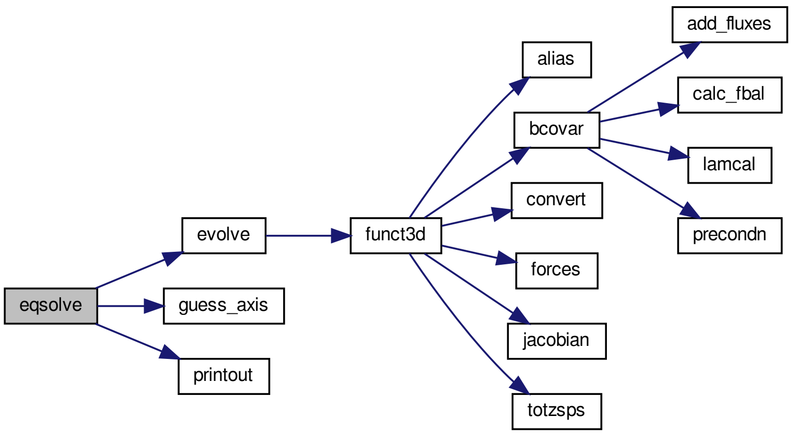
ier_flag	error flag

Definition at line 7 of file eqsolve.f90.

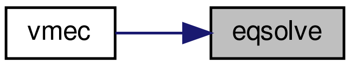
References evolve(), guess_axis(), and printout().

Referenced by vmec().

Here is the call graph for this function:



Here is the caller graph for this function:



7.30 src/evolve.f90 File Reference

Take a single time step in Fourier space to evolve the Fourier coefficients describing the equilibrium towards force balance.

Functions/Subroutines

- subroutine evolve (time_step, ier_flag, liter_flag)

  Take a single time step in Fourier space to evolve the Fourier coefficients describing the equilibrium towards force balance.

7.30.1 Detailed Description

Take a single time step in Fourier space to evolve the Fourier coefficients describing the equilibrium towards force balance.

7.30.2 Function/Subroutine Documentation

7.30.2.1 evolve()

```
subroutine evolve (
             real(rprec), intent(in) time_step,
             integer, intent(inout) ier_flag,
             logical, intent(inout) liter_flag )
```

Take a single time step in Fourier space to evolve the Fourier coefficients describing the equilibrium towards force balance.

Parameters

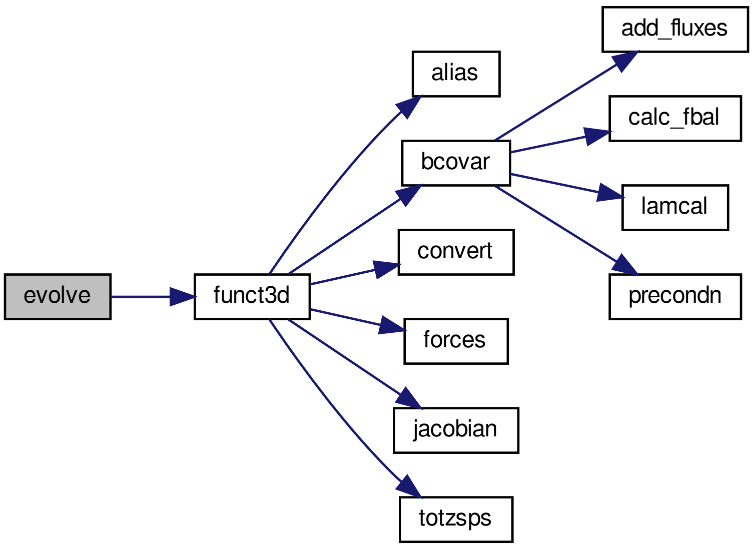
time_step	step length in parameter space to take
ier_flag	error flag
liter_flag	keep running?

Definition at line 11 of file evolve.f90.

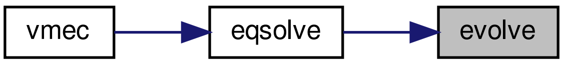
References funct3d().

Referenced by eqsolve().

Here is the call graph for this function:



Here is the caller graph for this function:



7.31 src/fileout.f90 File Reference

Write the output files.

Functions/Subroutines

- subroutine fileout (ier_flag)

    Write the output files.

7.31.1 Detailed Description

Write the output files.

7.31.2 Function/Subroutine Documentation

7.31.2.1 fileout()

```
subroutine fileout (
            integer, intent(inout) ier_flag )
```

Write the output files.

Parameters

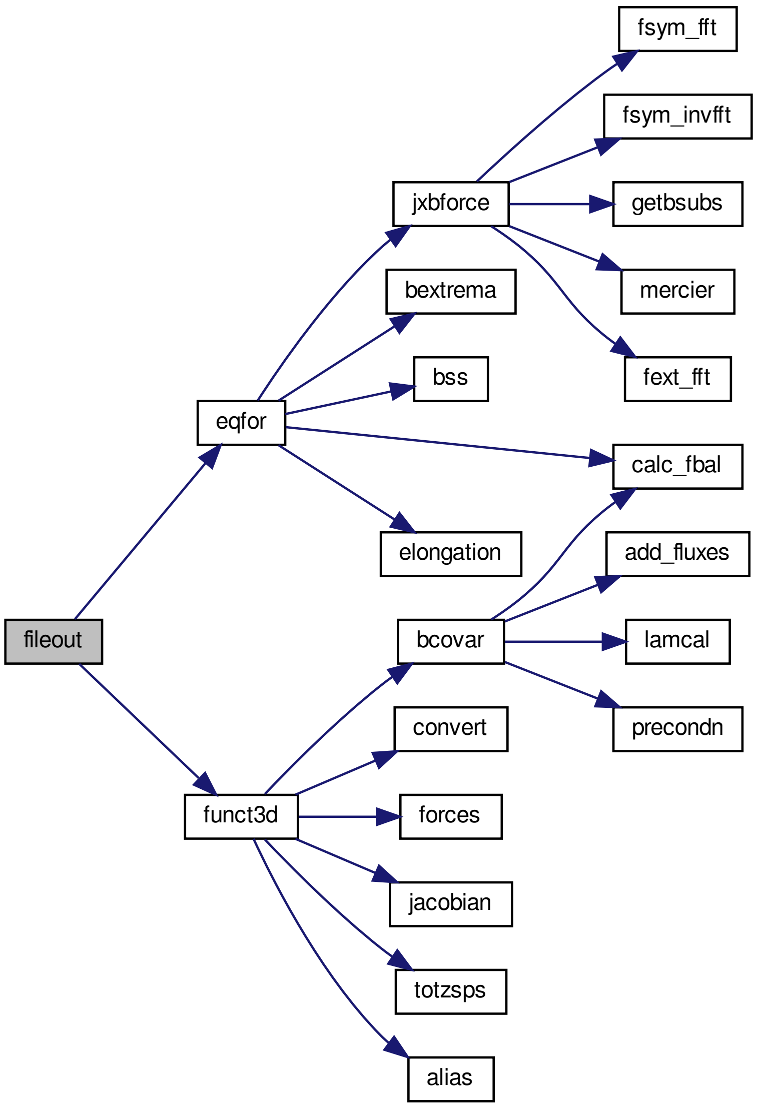
ier_flag	error flag

Definition at line 7 of file fileout.f90.

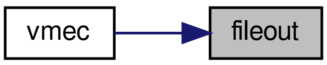
References eqfor(), and funct3d().

Referenced by vmec().

Here is the call graph for this function:



Here is the caller graph for this function:



7.32 src/fixaray.f90 File Reference

allocate and fill some fixed-size arrays (only depending on Fourier resolution).

Functions/Subroutines

- subroutine fixaray

  allocate and fill some fixed-size arrays (only depending on Fourier resolution).

7.32.1 Detailed Description

allocate and fill some fixed-size arrays (only depending on Fourier resolution).

7.33 src/flip_theta.f90 File Reference

Flip the definition of the poloidal angle in the user-provided initial guess for the LCFS geometry.

Functions/Subroutines

- subroutine flip_theta (rmn, zmn, lmn)

  Flip the definition of the poloidal angle in the user-provided initial guess for the LCFS geometry.

7.33.1 Detailed Description

Flip the definition of the poloidal angle in the user-provided initial guess for the LCFS geometry.

7.33.2 Function/Subroutine Documentation

7.33.2.1 flip_theta()

```
subroutine flip_theta (
            real(rprec), dimension(0:ntor,0:mpol1,ntmax), intent(inout) rmn,
            real(rprec), dimension(0:ntor,0:mpol1,ntmax), intent(inout) zmn,
            real(rprec), dimension(0:ntor,0:mpol1,ntmax), intent(inout), optional lmn )
```

Flip the definition of the poloidal angle in the user-provided initial guess for the LCFS geometry.

Parameters

	rmn	Fourier coefficients for R
	zmn	Fourier coefficients for Z
	lmn	Fourier coefficients for λ
`in,out`	*lmn*	never used: can also flip lambda...

Definition at line 9 of file flip_theta.f90.

7.34 src/forces.f90 File Reference

Compute the real-space MHD forces.

Functions/Subroutines

- subroutine forces

  Compute the real-space MHD forces.

7.34.1 Detailed Description

Compute the real-space MHD forces.

7.35 src/free_mem_funct3d.f90 File Reference

Free memory required by funct3d()

Functions/Subroutines

- subroutine free_mem_funct3d

  Free memory required by funct3d()

7.35.1 Detailed Description

Free memory required by funct3d()

7.36 src/free_mem_ns.f90 File Reference

Free memory depending on the number of flux surfaces `ns`.

Functions/Subroutines

- subroutine free_mem_ns

  Free memory depending on the number of flux surfaces `ns`.

7.36.1 Detailed Description

Free memory depending on the number of flux surfaces `ns`.

7.37 src/free_mem_nunv.f90 File Reference

Free arrays depending on the number of Fourier coefficients `nunv`.

Functions/Subroutines

- subroutine free_mem_nunv

  Free arrays depending on the number of Fourier coefficients `nunv`.

7.37.1 Detailed Description

Free arrays depending on the number of Fourier coefficients `nunv`.

7.38 src/freeb_data.f90 File Reference

Write out edge values of fields.

Functions/Subroutines

- subroutine freeb_data (rmnc, zmns, rmns, zmnc, bmodmn, bmodmn1)

  Write out edge values of fields.

7.38.1 Detailed Description

Write out edge values of fields.

7.38.2 Function/Subroutine Documentation

7.38.2.1 freeb_data()

```
subroutine freeb_data (
            real(rprec), dimension(mnmax) rmnc,
            real(rprec), dimension(mnmax) zmns,
            real(rprec), dimension(mnmax) rmns,
            real(rprec), dimension(mnmax) zmnc,
            real(rprec), dimension(mnmax) bmodmn,
            real(rprec), dimension(mnmax) bmodmn1 )
```

Write out edge values of fields.

Parameters

rmnc	stellarator-symmetric Fourier coefficients of R
zmns	stellarator-symmetric Fourier coefficients of Z
rmns	non-stellarator-symmetric Fourier coefficients of R
zmnc	non-stellarator-symmetric Fourier coefficients of Z
bmodmn	stellarator-symmetric Fourier coefficients of $\|\mathbf{B}\|$
bmodmn1	non-stellarator-symmetric Fourier coefficients of $\|\mathbf{B}\|$

Definition at line 12 of file freeb_data.f90.

7.39 src/fsym_fft.f90 File Reference

Fourier transforms.

Functions/Subroutines

- subroutine fext_fft (bout, bs_s, bs_a)

  Extends B_s from $\mathtt{ntheta2}$ interval to full $\mathtt{ntheta3}$ interval in angle θ.
- subroutine fsym_fft (bs, bu, bv, bs_s, bu_s, bv_s, bs_a, bu_a, bv_a)

  Contract bs,bu,bv from full $\mathtt{nu}$ interval to half-u interval so cos, sin integrals can be performed on half-u interval.

7.39.1 Detailed Description

Fourier transforms.

7.39.2 Function/Subroutine Documentation

7.39.2.1 fext_fft()

```
subroutine fext_fft (
            real(rprec), dimension(nzeta,ntheta3), intent(out) bout,
            real(rprec), dimension(nzeta,ntheta2), intent(in) bs_s,
            real(rprec), dimension(nzeta,ntheta2), intent(in) bs_a )
```

Extends B_s from $\mathtt{ntheta2}$ interval to full $\mathtt{ntheta3}$ interval in angle θ.

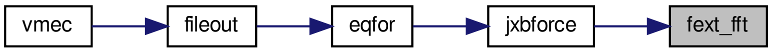
Parameters

bout	output B_s
bs↩_s	symmetric part of B_s
bs↩_a	anti-symmetric part of B_s

Definition at line 9 of file fsym_fft.f90.

Referenced by jxbforce().

Here is the caller graph for this function:



7.39.2.2 fsym_fft()

```
subroutine fsym_fft (
            real(rprec), dimension(nzeta,ntheta3), intent(in) bs,
            real(rprec), dimension(nzeta,ntheta3,0:1), intent(in) bu,
            real(rprec), dimension(nzeta,ntheta3,0:1), intent(in) bv,
            real(rprec), dimension(nzeta,ntheta2) bs_s,
            real(rprec), dimension(nzeta,ntheta2,0:1), intent(out) bu_s,
            real(rprec), dimension(nzeta,ntheta2,0:1), intent(out) bv_s,
            real(rprec), dimension(nzeta,ntheta2) bs_a,
            real(rprec), dimension(nzeta,ntheta2,0:1), intent(out) bu_a,
            real(rprec), dimension(nzeta,ntheta2,0:1), intent(out) bv_a )
```

Contract bs,bu,bv from full `nu` interval to half-u interval so cos, sin integrals can be performed on half-u interval.

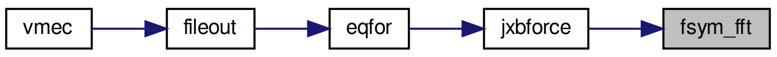
Parameters

bs	output B_s
bu	output B_θ
bv	output B_zeta
bs↵_s	symmetric part of B_s
bu↵_s	symmetric part of B_θ
bv↵_s	symmetric part of B_ζ
bs↵_a	anti-symmetric part of B_s
bu↵_a	anti-symmetric part of B_θ
bv↵_a	anti-symmetric part of B_ζ

Definition at line 47 of file fsym_fft.f90.

Referenced by jxbforce().

Here is the caller graph for this function:



7.40 src/fsym_invfft.f90 File Reference

Extends function from `ntheta2` to `ntheta3` range.

Functions/Subroutines

- subroutine fsym_invfft (bsubsu, bsubsv)

  Extends function from `ntheta2` to `ntheta3` range.

7.40.1 Detailed Description

Extends function from `ntheta2` to `ntheta3` range.

7.40.2 Function/Subroutine Documentation

7.40.2.1 fsym_invfft()

```
subroutine fsym_invfft (
            real(rprec), dimension(ns*nzeta,ntheta3,0:1), intent(inout) bsubsu,
            real(rprec), dimension(ns*nzeta,ntheta3,0:1), intent(inout) bsubsv )
```

Extends function from `ntheta2` to `ntheta3` range.

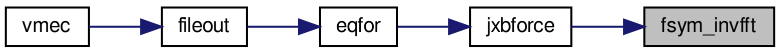
Parameters

bsubsu	tangential derivative of covariant magnetic field component $\partial B_s / \partial \theta$
bsubsv	tangential derivative of covariant magnetic field component $\partial B_s / \partial \zeta$

Definition at line 8 of file fsym_invfft.f90.

Referenced by jxbforce().

Here is the caller graph for this function:



7.41 src/funct3d.f90 File Reference

Evaluate the three-dimensional MHD energy functional.

Functions/Subroutines

- subroutine funct3d (ier_flag)

  Evaluate the three-dimensional MHD energy functional.

7.41.1 Detailed Description

Evaluate the three-dimensional MHD energy functional.

7.41.2 Function/Subroutine Documentation

7.41.2.1 funct3d()

```
subroutine funct3d (
            integer, intent(inout) ier_flag )
```

Evaluate the three-dimensional MHD energy functional.

Parameters

ier_flag	error flag

use system call to stand-alone NESTOR for vacuum computation

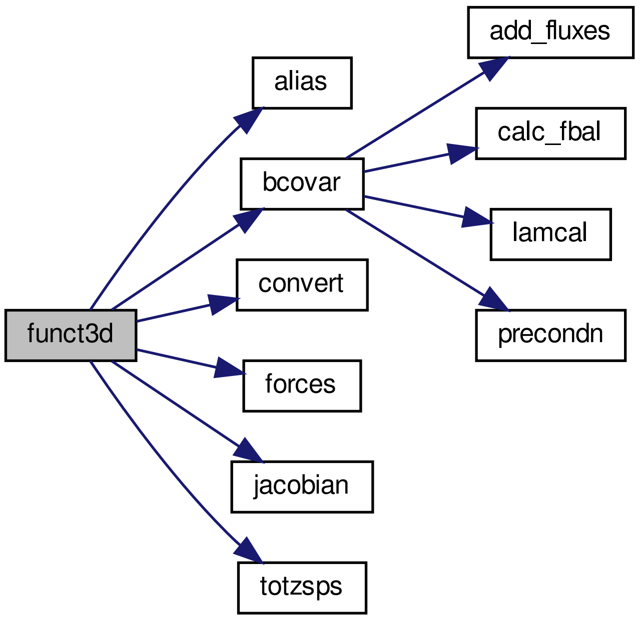
dump reference input for and output of NESTOR when using internal NESTOR

Definition at line 7 of file funct3d.f90.

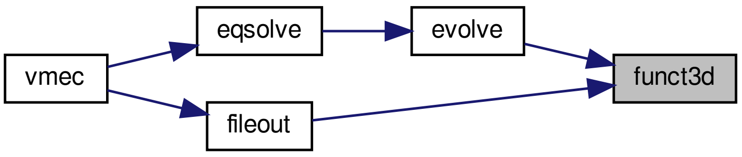
References alias(), bcovar(), convert(), forces(), jacobian(), and totzsps().

Referenced by evolve(), and fileout().

Here is the call graph for this function:



Here is the caller graph for this function:



7.42 src/functions.f File Reference

This module containes functions used by the profiles.

Functions/Subroutines

- real(rprec) function, public functions::two_power (x, b)

  Profile function for the `two_power` profile. $b(0) * (1 - x^{b(1)})^{b(2)}$.
- real(rprec) function, public functions::two_power_gs (x, b)

  Profile function for the `two_power_gs` profile. $two\_power(x)*(1+\sum\left[b(i)*\exp(-(x-b(i+1))/b(i+2))^2\right])$.
- logical function functions::function_test ()

  Main test function.

7.42.1 Detailed Description

This module containes functions used by the profiles.

7.42.2 Function/Subroutine Documentation

7.42.2.1 function_test()

```
logical function functions::function_test
```

Main test function.

Test `two_power` function for x = 0, b = {1,10,2} is 1

Test `two_power` function for x = 1, b = {1,10,2} is 0

Test `two_power` function for x = 0.5, b = {1,1,1} is 0.5

Test `two_power` function for x = 0.5, b = {1,1,2} is 0.25

Test `two_power_gs` function for x = 0.4, b = {1,1,1,0,0,1} is two_power(x,b)

Test `two_power_gs` function for x = 0.8, b = {1,1,0,1,0.8,0.1} is 2

Definition at line 51 of file functions.f.

7.42.2.2 two_power()

```
real(rprec) function, public functions::two_power (
            real(rprec), intent(in) x,
            real(rprec), dimension(0:20), intent(in) b )
```

Profile function for the `two_power` profile. $b(0) * (1 - x^{b(1)})^{b(2)}$.

Parameters

x	evaluation location
b	parameter vector

Definition at line 20 of file functions.f.

7.42.2.3 two_power_gs()

```
real(rprec) function, public functions::two_power_gs (
            real(rprec), intent(in) x,
            real(rprec), dimension(0:20), intent(in) b )
```

Profile function for the `two_power_gs` profile. $\mathtt{two\_power}(x)*(1+\sum\left[b(i)*\exp(-(x-b(i+1))/b(i+2))^2\right])$.

Parameters

x	evaluation location
b	parameter vector

Definition at line 34 of file functions.f.

7.43 src/getbsubs.f90 File Reference

Solves the radial force balance $\mathbf{B}\cdot B_s = F_s$ for B_s in real space using collocation.

Functions/Subroutines

- subroutine getbsubs (bsubsmn, frho, bsupu, bsupv, mmax, nmax, info)

  Solves the radial force balance $\mathbf{B}\cdot B_s = F_s$ for B_s in real space using collocation.

7.43.1 Detailed Description

Solves the radial force balance $\mathbf{B}\cdot B_s = F_s$ for B_s in real space using collocation.

7.43.2 Function/Subroutine Documentation

7.43.2.1 getbsubs()

```
subroutine getbsubs (
            real(rprec), dimension(0:mmax, -nmax:nmax, 0:1), intent(out) bsubsmn,
            real(rprec), dimension(nzeta, ntheta3), intent(in) frho,
            real(rprec), dimension(nzeta, ntheta3), intent(in) bsupu,
            real(rprec), dimension(nzeta, ntheta3), intent(in) bsupv,
            integer, intent(in) mmax,
            integer, intent(in) nmax,
            integer, intent(out) info )
```

Solves the radial force balance $\mathbf{B} \cdot B_s = F_s$ for B_s in real space using collocation.

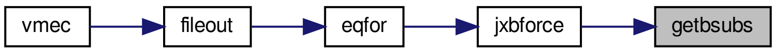
Parameters

bsubsmn	Fourier coefficients of B_s
frho	Fourier coefficients of radial Force component
bsupu	contravariant component of magnetic field B^θ
bsupv	contravariant component of magnetic field B^ζ
mmax	maximum poloidal mode number
nmax	maximum toroidal mode number
info	error flag

Definition at line 13 of file getbsubs.f90.

Referenced by jxbforce().

Here is the caller graph for this function:



7.44 src/getcurmid.f90 File Reference

Get current at midplane (?)

Functions/Subroutines

- subroutine getcurmid (curmid, izeta, gsqrt, r12)

  Get current at midplane (?)

7.44.1 Detailed Description

Get current at midplane (?)

7.44.2 Function/Subroutine Documentation

7.44.2.1 getcurmid()

```
subroutine getcurmid (
             real(rprec), dimension(2*ns) curmid,
             real(rprec), dimension(ns,nzeta,*) izeta,
             real(rprec), dimension(ns,nzeta,*) gsqrt,
             real(rprec), dimension(ns,nzeta,*) r12 )
```

Get current at midplane (?)

Parameters

curmid	current at midplane (?)
izeta	index in toroidal direction
gsqrt	Jacobian
r12	R^2

Definition at line 10 of file getcurmid.f90.

7.45 src/getfsq.f90 File Reference

Compute total force residual on flux surfaces.

Functions/Subroutines

- subroutine getfsq (gcr, gcz, gnormr, gnormz, gnorm, medge)
  Compute total force residual on flux surfaces.

7.45.1 Detailed Description

Compute total force residual on flux surfaces.

7.45.2 Function/Subroutine Documentation

7.45.2.1 getfsq()

```
subroutine getfsq (
             real(rprec), dimension(ns,mnsize*ntmax), intent(in) gcr,
             real(rprec), dimension(ns,mnsize*ntmax), intent(in) gcz,
             real(rprec), intent(out) gnormr,
             real(rprec), intent(out) gnormz,
             real(rprec), intent(in) gnorm,
             integer, intent(in) medge )
```

Compute total force residual on flux surfaces.

Parameters

gcr	R-component of force
gcz	Z-component of force
gnormr	normalized total force residual in R
gnormz	normalized total force residual in Z
gnorm	normalization factor for forces
medge	=0: exclude contribution from LCFS; =1: include LCFS contribution

Definition at line 12 of file getfsq.f90.

7.46 src/guess_axis.f90 File Reference

Computes guess for magnetic axis if user guess leads to initial sign change of Jacobian.

Functions/Subroutines

- subroutine guess_axis (r1, z1, ru0, zu0)
  Computes guess for magnetic axis if user guess leads to initial sign change of Jacobian.

7.46.1 Detailed Description

Computes guess for magnetic axis if user guess leads to initial sign change of Jacobian.

7.46.2 Function/Subroutine Documentation

7.46.2.1 guess_axis()

```
subroutine guess_axis (
             real(rprec), dimension(ns,nzeta,ntheta3,0:1), intent(in) r1,
             real(rprec), dimension(ns,nzeta,ntheta3,0:1), intent(in) z1,
             real(rprec), dimension(ns,nzeta,ntheta3), intent(in) ru0,
             real(rprec), dimension(ns,nzeta,ntheta3), intent(in) zu0 )
```

Computes guess for magnetic axis if user guess leads to initial sign change of Jacobian.

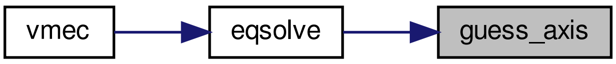
Parameters

r1	R
z1	Z
ru0	$\partial R/\partial\theta$
zu0	$\partial Z/\partial\theta$

Definition at line 10 of file guess_axis.f90.

Referenced by eqsolve().

Here is the caller graph for this function:



7.47 src/heading.f90 File Reference

Open output files and print banner message at the top.

Functions/Subroutines

- subroutine heading (extension)

  Open output files and print banner message at the top.

7.47.1 Detailed Description

Open output files and print banner message at the top.

7.47.2 Function/Subroutine Documentation

7.47.2.1 heading()

```
subroutine heading (
             character(len=*), intent(in) extension )
```

Open output files and print banner message at the top.

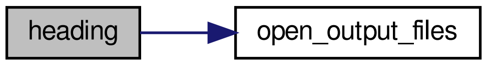
Parameters

extension	input file "extension": part after `'input.'`.

Definition at line 7 of file heading.f90.

References open_output_files().

Here is the call graph for this function:



7.48 src/initialize_radial.f90 File Reference

Allocates memory for radial arrays and initializes radial profiles.

Functions/Subroutines

- subroutine initialize_radial (nsval, ns_old, delt0)

  Allocates memory for radial arrays and initializes radial profiles.

7.48.1 Detailed Description

Allocates memory for radial arrays and initializes radial profiles.

7.48.2 Function/Subroutine Documentation

7.48.2.1 initialize_radial()

```
subroutine initialize_radial (
            integer, intent(in) nsval,
            integer, intent(inout) ns_old,
            real(rprec), intent(out) delt0 )
```

Allocates memory for radial arrays and initializes radial profiles.

Parameters

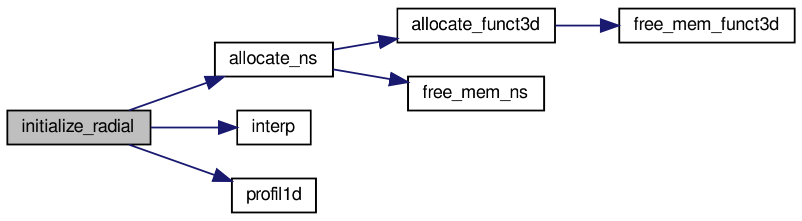
nsval	new number of flux surfaces
ns_old	old number of flux surfaces (from previous multi-grid iteration)
delt0	time step to be used in the new multi-grid iteration

Definition at line 9 of file initialize_radial.f90.

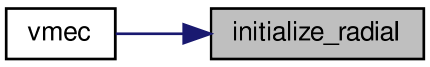
References allocate_ns(), interp(), and profil1d().

Referenced by vmec().

Here is the call graph for this function:



Here is the caller graph for this function:



7.49 src/interp.f90 File Reference

Interpolate R, Z and $lambda$ on full grid.

Functions/Subroutines

- subroutine interp (xnew, xold, scalxc, nsnew, nsold)
  Interpolate R, Z and $lambda$ on full grid.

7.49.1 Detailed Description

Interpolate R, Z and $lambda$ on full grid.

7.49.2 Function/Subroutine Documentation

7.49.2.1 interp()

```
subroutine interp (
            real(rprec), dimension(nsnew,mnsize,3*ntmax), intent(out) xnew,
            real(rprec), dimension(nsold,mnsize,3*ntmax), intent(inout) xold,
            real(rprec), dimension(nsnew,mnsize,3*ntmax), intent(in) scalxc,
            integer, intent(in) nsnew,
            integer, intent(in) nsold )
```

Interpolate R, Z and $lambda$ on full grid.

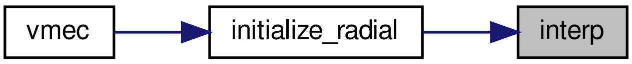
Parameters

xnew	interpolated state vector (nsnew surfaces)
xold	interpolation basis: old state vector (nsold surfaces)
scalxc	scaling factors to normalize the new state vector to
nsnew	new number of flux surfaces
nsold	old number of flux surfaces

Definition at line 11 of file interp.f90.

Referenced by initialize_radial().

Here is the caller graph for this function:



7.50 src/jacobian.f90 File Reference

Evaulate the Jacobian of the transform from flux- to cylindrical coordinates.

Functions/Subroutines

- subroutine jacobian

  Evaulate the Jacobian of the transform from flux- to cylindrical coordinates.

7.50.1 Detailed Description

Evaulate the Jacobian of the transform from flux- to cylindrical coordinates.

7.51 src/jxbforce.f90 File Reference

Program for computing local $\mathbf{K} \times \mathbf{B} = \nabla p$ force balance.

Functions/Subroutines

- subroutine jxbforce (bsupu, bsupv, bsubu, bsubv, bsubsh, bsubsu, bsubsv, gsqrt, bsq, itheta, izeta, brho, ier_flag)

  Program for computing local $\mathbf{K} \times \mathbf{B} = \nabla p$ force balance.

7.51.1 Detailed Description

Program for computing local $\mathbf{K} \times \mathbf{B} = \nabla p$ force balance.

7.51.2 Function/Subroutine Documentation

7.51.2.1 jxbforce()

```
subroutine jxbforce (
              real(rprec), dimension(ns,nznt), intent(in) bsupu,
              real(rprec), dimension(ns,nznt), intent(in) bsupv,
              real(rprec), dimension(ns,nznt,0:1), intent(inout), target bsubu,
              real(rprec), dimension(ns,nznt,0:1), intent(inout), target bsubv,
              real(rprec), dimension(ns,nznt), intent(in) bsubsh,
              real(rprec), dimension(ns,nznt,0:1) bsubsu,
              real(rprec), dimension(ns,nznt,0:1) bsubsv,
              real(rprec), dimension(ns,nznt), intent(in) gsqrt,
              real(rprec), dimension(ns,nznt), intent(in) bsq,
              real(rprec), dimension(ns,nznt), intent(out) itheta,
              real(rprec), dimension(ns,nznt), intent(out) izeta,
              real(rprec), dimension(ns,nznt), intent(out) brho,
              integer, intent(in) ier_flag )
```

Program for computing local $\mathbf{K} \times \mathbf{B} = \nabla p$ force balance.

Parameters

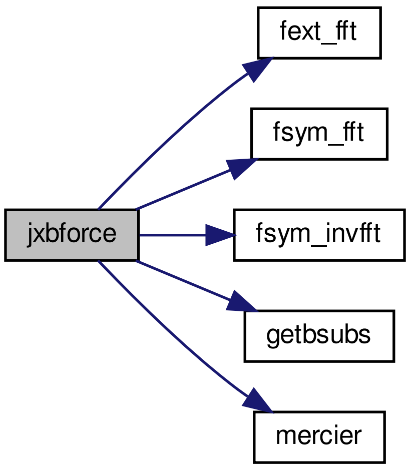
bsupu	contravariant component of magnetic field B^θ
bsupv	contravariant component of magnetic field B^ζ
bsubu	covariant component of magnetic field B_θ
bsubv	covariant component of magnetic field B_ζ
bsubsh	covariant component of magnetic field B_s (on half grid?)
bsubsu	tangential derivate of covariant component of magnetic field $\partial B_s / \partial \theta$ (?)
bsubsv	tangential derivate of covariant component of magnetic field $\partial B_s / \partial \zeta$ (?)
gsqrt	Jacobian $\sqrt{g}$
bsq	modulus of magnetic field $\|\mathbf{B}\|^2$
itheta	index in poloidal direction
izeta	index in toroidal direction
brho	radial component of magnetic field B_ρ (?)
ier_flag	error flag

Definition at line 19 of file jxbforce.f90.

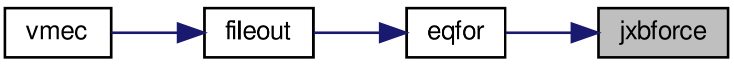
References fext_fft(), fsym_fft(), fsym_invfft(), getbsubs(), and mercier().

Referenced by eqfor().

Here is the call graph for this function:



Here is the caller graph for this function:



7.52 src/lamcal.f90 File Reference

Normalization parameters for λ.

Functions/Subroutines

- subroutine lamcal (overg, guu, guv, gvv)

  Normalization parameters for λ.

7.52.1 Detailed Description

Normalization parameters for λ.

7.52.2 Function/Subroutine Documentation

7.52.2.1 lamcal()

```
subroutine lamcal (
            real(rprec), dimension(ns,nznt), intent(in) overg,
            real(rprec), dimension(ns,nznt), intent(in) guu,
            real(rprec), dimension(ns,nznt), intent(in) guv,
            real(rprec), dimension(ns,nznt), intent(in) gvv )
```

Normalization parameters for λ.

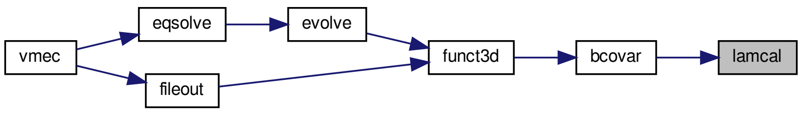
Parameters

overg	inverse of Jacobian $1/\sqrt{g}$
guu	metric element $g_{\theta\theta}$
guv	metric element $g_{\theta\zeta}$
gvv	metric element $g_{\zeta\zeta}$

Definition at line 10 of file lamcal.f90.

Referenced by bcovar().

Here is the caller graph for this function:



7.53 src/line_segment.f File Reference

This module containes code to create a profile constructed of line segments.

Modules

- module line_segment

  This module containes code to create a profile constructed of line segments. These line segments are assumed to be specified such that $xx(i) < xx(i + 1)$.

Functions/Subroutines

- subroutine, public **line_segment::line_seg** (x, y, xx, yy, n)
- subroutine, public **line_segment::line_seg_int** (x, y, xx, yy, n)
- logical function, public **line_segment::line_seg_test** ()

7.53.1 Detailed Description

This module containes code to create a profile constructed of line segments.

7.54 src/magnetic_fluxes.f90 File Reference

Compute toroidal and poloidal magnetic flux profiles.

Functions/Subroutines

- real(rprec) function torflux_deriv (x)

  Compute the radial derivative of the enclosed toroidal magnetic flux.
- real(rprec) function polflux_deriv (x)

  Compute the radial derivative of the enclosed poloidal magnetic flux.
- real(rprec) function torflux (x)

  Compute the enclosed toroidal magnetic flux.
- real(rprec) function polflux (x)

  Compute the enclosed poloidal magnetic flux.

7.54.1 Detailed Description

Compute toroidal and poloidal magnetic flux profiles.

7.54.2 Function/Subroutine Documentation

7.54.2.1 polflux()

```
real(rprec) function polflux (
            real(rprec), intent(in) x )
```

Compute the enclosed poloidal magnetic flux.

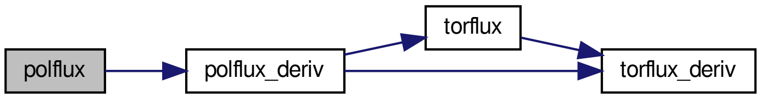
Parameters

	x	evaluation location
in	*x*	radial flux variable (=TOROIDAL FLUX ONLY IF APHI=1)

Definition at line 75 of file magnetic_fluxes.f90.

References polflux_deriv().

Here is the call graph for this function:



7.54.2.2 polflux_deriv()

```
real(rprec) function polflux_deriv (
             real(rprec), intent(in) x )
```

Compute the radial derivative of the enclosed poloidal magnetic flux.

Parameters

	x	evaluation location
in	*x*	radial flux variable (=TOROIDAL FLUX ONLY IF APHI=1)

Returns

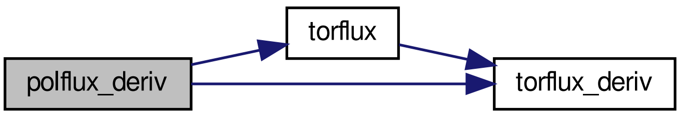
polflux_deriv == d(chi)/dx = iota(TF(x)) $*$ torflux_deriv(x)

Definition at line 28 of file magnetic_fluxes.f90.

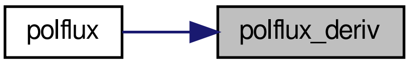
References torflux(), and torflux_deriv().

Referenced by polflux().

Here is the call graph for this function:



Here is the caller graph for this function:



7.54.2.3 torflux()

```
real(rprec) function torflux (
             real(rprec), intent(in) x )
```

Compute the enclosed toroidal magnetic flux.

Parameters

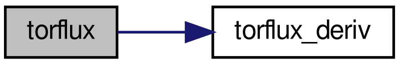
	x	evaluation location
in	*x*	radial flux variable (=TOROIDAL FLUX ONLY IF APHI=1)

Definition at line 51 of file magnetic_fluxes.f90.

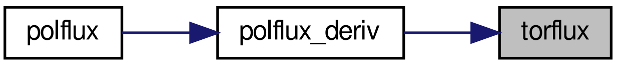
References torflux_deriv().

Referenced by polflux_deriv().

Here is the call graph for this function:



Here is the caller graph for this function:



7.54.2.4 torflux_deriv()

```
real(rprec) function torflux_deriv (
            real(rprec), intent(in) x )
```

Compute the radial derivative of the enclosed toroidal magnetic flux.

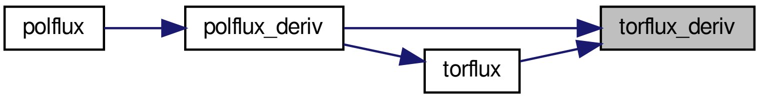
Parameters

	x	evaluation location
in	*x*	radial flux variable (=TOROIDAL FLUX ONLY IF APHI=1)

Definition at line 7 of file magnetic_fluxes.f90.

Referenced by polflux_deriv(), and torflux().

Here is the caller graph for this function:



7.55 src/mercier.f90 File Reference

Evaluate the Mercier stability criterion.

Functions/Subroutines

- subroutine mercier (gsqrt, bsq, bdotj, iotas, wint, r1, rt, rz, zt, zz, bsubu, vp, phips, pres, ns, nznt)

  Evaluate the Mercier stability criterion.

7.55.1 Detailed Description

Evaluate the Mercier stability criterion.

7.55.2 Function/Subroutine Documentation

7.55.2.1 mercier()

```
subroutine mercier (
            real(rprec), dimension(ns,nznt), intent(in) gsqrt,
            real(rprec), dimension(ns,nznt), intent(in) bsq,
            real(rprec), dimension(ns,nznt), intent(inout) bdotj,
            real(rprec), dimension(ns), intent(in) iotas,
            real(rprec), dimension(ns*nznt), intent(in) wint,
            real(rprec), dimension(ns,nznt,0:1), intent(in) r1,
            real(rprec), dimension(ns,nznt,0:1), intent(in) rt,
            real(rprec), dimension(ns,nznt,0:1), intent(in) rz,
            real(rprec), dimension(ns,nznt,0:1), intent(in) zt,
            real(rprec), dimension(ns,nznt,0:1), intent(in) zz,
            real(rprec), dimension(ns*nznt), intent(in) bsubu,
            real(rprec), dimension(ns), intent(in) vp,
            real(rprec), dimension(ns), intent(in) phips,
            real(rprec), dimension(ns), intent(in) pres,
            integer, intent(in) ns,
            integer, intent(in) nznt )
```

Evaluate the Mercier stability criterion.

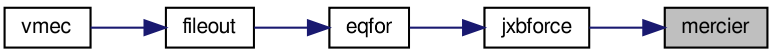
Parameters

gsqrt	Jacobian $\sqrt{g}$
bsq	modulus of magnetic field $|\mathbf{B}|$
bdotj	parallel current density $\mathbf{B}\cdot\mathbf{j}$
iotas	rotational transform profile
wint	normalization constant for flux-surface integrals
r1	R
rt	$\partial R/\partial\theta$
rz	$\partial R/\partial\zeta$
zt	$\partial Z/\partial\theta$
zz	$\partial Z/\partial\zeta$
bsubu	contravariant component of magnetic field B^ζ
vp	radial profile of specific volume $\partial V/\partial s$
phips	radial derivative of enclosed toroidal magnetic flux
pres	pressure profile
ns	number of flux surfaces
nznt	number of grid points per flux surface

Definition at line 22 of file mercier.f90.

Referenced by jxbforce().

Here is the caller graph for this function:



7.56 src/mgrid_mod.f File Reference

Precomputed table of magnetic field due to confimenent coils.

Modules

- module mgrid_mod

  Precomputed table of magnetic field due to confimenent coils.

Functions/Subroutines

- subroutine **mgrid_mod::read_mgrid** (mgrid_file, extcur, nv, nfp, lscreen, ier_flag)
- subroutine **mgrid_mod::sum_bfield** (bfield, bf_add, cur, n1)
- subroutine **mgrid_mod::assign_bptrs** (bptr)
- subroutine **mgrid_mod::free_mgrid** (istat)

Variables

- integer, parameter **mgrid_mod::nlimset** = 2
- character(len= $*$), parameter **mgrid_mod::vn_br0** = 'br'
- character(len= $*$), parameter **mgrid_mod::vn_bp0** = 'bp'
- character(len= $*$), parameter **mgrid_mod::vn_bz0** = 'bz'
- character(len= $*$), parameter **mgrid_mod::vn_ir** = 'ir'
- character(len= $*$), parameter **mgrid_mod::vn_jz** = 'jz'
- character(len= $*$), parameter **mgrid_mod::vn_kp** = 'kp'
- character(len= $*$), parameter **mgrid_mod::vn_nfp** = 'nfp'
- character(len= $*$), parameter **mgrid_mod::vn_rmin** ='rmin'
- character(len= $*$), parameter **mgrid_mod::vn_rmax** ='rmax'
- character(len= $*$), parameter **mgrid_mod::vn_zmin** ='zmin'
- character(len= $*$), parameter **mgrid_mod::vn_zmax** ='zmax'
- character(len= $*$), parameter **mgrid_mod::vn_coilgrp** ='coil_group'
- character(len= $*$), parameter **mgrid_mod::vn_nextcur** = 'nextcur'
- character(len= $*$), parameter **mgrid_mod::vn_mgmode** ='mgrid_mode'
- character(len= $*$), parameter **mgrid_mod::vn_coilcur** = 'raw_coil_cur'
- character(len= $*$), parameter **mgrid_mod::ln_next** = 'External currents'
- integer **mgrid_mod::nr0b**
- integer **mgrid_mod::np0b**
- integer **mgrid_mod::nfper0**
- integer **mgrid_mod::nz0b**
- integer **mgrid_mod::nobd**
- integer **mgrid_mod::nobser**
- integer **mgrid_mod::nextcur**
- integer **mgrid_mod::nbfldn**
- integer **mgrid_mod::nbsets**
- integer **mgrid_mod::nbcoilsn**
- integer **mgrid_mod::nbvac**
- integer **mgrid_mod::nbcoil_max**
- integer **mgrid_mod::nlim**
- integer **mgrid_mod::nlim_max**
- integer **mgrid_mod::nsets**
- integer **mgrid_mod::nrgrid**

- integer **mgrid_mod::nzgrid**
- integer, dimension(:), allocatable **mgrid_mod::needflx**
- integer, dimension(:), allocatable **mgrid_mod::nbcoils**
- integer, dimension(:), allocatable **mgrid_mod::limitr**
- integer, dimension(:), allocatable **mgrid_mod::nsetsn**
- integer, dimension(:,:), allocatable **mgrid_mod::iconnect**
- integer, dimension(:,:), allocatable **mgrid_mod::needbfld**
- real(rprec) **mgrid_mod::rminb**
- real(rprec) **mgrid_mod::zminb**
- real(rprec) **mgrid_mod::rmaxb**
- real(rprec) **mgrid_mod::zmaxb**
- real(rprec) **mgrid_mod::delrb**
- real(rprec) **mgrid_mod::delzb**
- real(rprec) **mgrid_mod::rx1**
- real(rprec) **mgrid_mod::rx2**
- real(rprec) **mgrid_mod::zy1**
- real(rprec) **mgrid_mod::zy2**
- real(rprec) **mgrid_mod::condif**
- real(rprec), dimension(:,:), allocatable, target **mgrid_mod::bvac**
- real(rprec), dimension(:,:,:), pointer **mgrid_mod::brvac**
- real(rprec), dimension(:,:,:), pointer **mgrid_mod::bzvac**
- real(rprec), dimension(:,:,:), pointer **mgrid_mod::bpvac**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::unpsiext**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::plbfld**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::rbcoil**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::zbcoil**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::abcoil**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::bcoil**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::rbcoilsqr**
- real(rprec), dimension(:), allocatable **mgrid_mod::raw_coil_current**
- real(rprec), dimension(:), allocatable **mgrid_mod::xobser**
- real(rprec), dimension(:), allocatable **mgrid_mod::zobser**
- real(rprec), dimension(:), allocatable **mgrid_mod::xobsqr**
- real(rprec), dimension(:), allocatable **mgrid_mod::dsiext**
- real(rprec), dimension(:), allocatable **mgrid_mod::psiext**
- real(rprec), dimension(:), allocatable **mgrid_mod::plflux**
- real(rprec), dimension(:), allocatable **mgrid_mod::b_chi**
- character(len=300) **mgrid_mod::mgrid_path**
- character(len=300) **mgrid_mod::mgrid_path_old** = " "
- character(len=30), dimension(:), allocatable **mgrid_mod::curlabel**
- character(len=15), dimension(:), allocatable **mgrid_mod::dsilabel**
- character(len=15), dimension(:), allocatable **mgrid_mod::bloopnames**
- character(len=30) **mgrid_mod::tokid**
- real(rprec), dimension(:,:,:), allocatable **mgrid_mod::dbcoil**
- real(rprec), dimension(:,:,:), allocatable **mgrid_mod::pfcspec**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::rlim**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::zlim**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::reslim**
- real(rprec), dimension(:,:), allocatable **mgrid_mod::seplim**
- character(len=1) **mgrid_mod::mgrid_mode**

7.56.1 Detailed Description

Precomputed table of magnetic field due to confimenent coils.

7.57 src/NESTOR/analysum.f90 File Reference

Functions/Subroutines

- subroutine **analysum** (grpmn, bvec, sl, tl, m, n, l, ivacskip, lasym, m_map, n_map, grpmn_m_map, grpmn←
_n_map)

7.58 src/NESTOR/analysum2.f90 File Reference

Functions/Subroutines

- subroutine **analysum2** (grpmn, bvec, m, n, l, ivacskip, lasym, m_map, n_map, grpmn_m_map, grpmn_n_←
map)

7.59 src/NESTOR/analyt.f90 File Reference

Functions/Subroutines

- subroutine **analyt** (grpmn, bvec, ivacskip, lasym, m_map, n_map, grpmn_m_map, grpmn_n_map)

7.60 src/NESTOR/becoil.f90 File Reference

Functions/Subroutines

- subroutine **becoil** (rad, zee, brvac, bpvac, bzvac)

7.61 src/NESTOR/belicu.f90 File Reference

Functions/Subroutines

- subroutine **belicu** (torcur, bx, by, bz, cos1, sin1, rp, zp)

7.62 src/NESTOR/bextern.f90 File Reference

Functions/Subroutines

- subroutine **bextern** (plascur, wint)

7.63 src/NESTOR/data/nestor_io.f90 File Reference

Input and Output for stand-alone NESTOR.

Modules

- module nestor_io

    Input and Output for stand-alone NESTOR.

Functions/Subroutines

- subroutine **nestor_io::read_nestor_inputs** (vac_file)
- subroutine **nestor_io::write_nestor_outputs** (vac_file, lasym, ivac, ier_flag)
- subroutine **write_nestor_inputs** (vac_file, vacuum_calls, ier_flag, mgrid_file, input_extension, ivacskip, ivac, nfp, ntor, mpol, nzeta, ntheta, mnmax, xm, xn, rmnc, zmns, rmns, zmnc, rbtor, ctor, lasym, signgs, extcur_↩ nestor, raxis_nestor, zaxis_nestor, wint, nznt, amatsav, bvecsav, mnpd2, bsubvvac)
- subroutine **read_nestor_outputs** (vac_file, ier_flag, ivac)

Variables

- character(len=255) **nestor_io::input_extension**
- character(len=255) **nestor_io::mgrid_file**
- real(dp), dimension(:), allocatable **nestor_io::extcur**
- real(dp), dimension(:), allocatable **nestor_io::raxis**
- real(dp), dimension(:), allocatable **nestor_io::zaxis**
- real(dp), dimension(:), allocatable **nestor_io::xm**
- real(dp), dimension(:), allocatable **nestor_io::xn**
- real(dp), dimension(:), allocatable **nestor_io::rmnc**
- real(dp), dimension(:), allocatable **nestor_io::zmns**
- real(dp), dimension(:), allocatable **nestor_io::rmns**
- real(dp), dimension(:), allocatable **nestor_io::zmnc**
- real(dp), dimension(:), allocatable **nestor_io::wint**
- integer **nestor_io::nfp**
- integer **nestor_io::ntor**
- integer **nestor_io::mpol**
- integer **nestor_io::ntheta**
- integer **nestor_io::nzeta**
- integer **nestor_io::nextcur**
- integer **nestor_io::ier_flag**
- integer **nestor_io::ivac**
- integer **nestor_io::ivacskip**
- integer **nestor_io::mnmax**
- integer **nestor_io::vacuum_calls**
- logical **nestor_io::lasym**
- real(dp) **nestor_io::ctor**
- real(dp) **nestor_io::rbtor**
- real(dp) **nestor_io::signgs**
- integer **nestor_io::mnpd2_nestor**
- real(dp), dimension(:), allocatable **nestor_io::amatsav_nestor**
- real(dp), dimension(:), allocatable **nestor_io::bvecsav_nestor**
- real(dp) **nestor_io::bsubvvac_nestor**
- character(len= ∗), dimension(1), parameter **nestor_io::mn1dim** = (/'mn_mode'/)
- character(len= ∗), dimension(1), parameter **nestor_io::mnpotdim** = (/'mn_mode_pot'/)
- character(len= ∗), dimension(1), parameter **nestor_io::nzntdim** = (/'nznt'/)
- character(len= ∗), dimension(1), parameter **nestor_io::nzetadim** = (/'nzeta'/)
- character(len= ∗), dimension(1), parameter **nestor_io::nextcurim** = (/'nextcur'/)
- character(len= ∗), dimension(1), parameter **nestor_io::bvecsavdim** =(/'mnpd2'/)

- character(len= ∗), dimension(1), parameter **nestor_io::amatsavdim** =(/'mnpd2_times_mnpd2'/)
- character(len= ∗), dimension(2), parameter **nestor_io::r2dim** = (/'mn_mode','radius '/)
- character(len= ∗), parameter **nestor_io::vn_vacuum_calls** = 'vacuum_calls'
- character(len= ∗), parameter **nestor_io::vn_ier_flag** = "ier_flag"
- character(len= ∗), parameter **nestor_io::vn_mgrid** = "mgrid_file"
- character(len= ∗), parameter **nestor_io::vn_inputext** = "input_extension"
- character(len= ∗), parameter **nestor_io::vn_ivacskip** = "ivacskip"
- character(len= ∗), parameter **nestor_io::vn_ivac** = "ivac"
- character(len= ∗), parameter **nestor_io::vn_nfp** = "nfp"
- character(len= ∗), parameter **nestor_io::vn_ntor** = "ntor"
- character(len= ∗), parameter **nestor_io::vn_mpol** = "mpol"
- character(len= ∗), parameter **nestor_io::vn_nzeta** = "nzeta"
- character(len= ∗), parameter **nestor_io::vn_ntheta** = "ntheta"
- character(len= ∗), parameter **nestor_io::vn_mnmax** = "mnmax"
- character(len= ∗), parameter **nestor_io::vn_pmod** = "xm"
- character(len= ∗), parameter **nestor_io::vn_tmod** = "xn"
- character(len= ∗), parameter **nestor_io::vn_rmnc** = "rmnc"
- character(len= ∗), parameter **nestor_io::vn_zmns** = "zmns"
- character(len= ∗), parameter **nestor_io::vn_rmns** = "rmns"
- character(len= ∗), parameter **nestor_io::vn_zmnc** = "zmnc"
- character(len= ∗), parameter **nestor_io::vn_rbtor** = "rbtor"
- character(len= ∗), parameter **nestor_io::vn_ctor** = "ctor"
- character(len= ∗), parameter **nestor_io::vn_lasym** = "lasym"
- character(len= ∗), parameter **nestor_io::vn_signgs** = "signgs"
- character(len= ∗), parameter **nestor_io::vn_extcur** = "extcur"
- character(len= ∗), parameter **nestor_io::vn_raxis_nestor** = "raxis_nestor"
- character(len= ∗), parameter **nestor_io::vn_zaxis_nestor** = "zaxis_nestor"
- character(len= ∗), parameter **nestor_io::vn_wint** = "wint"
- character(len= ∗), parameter **nestor_io::vn_bsqvac** = "bsqvac"
- character(len= ∗), parameter **nestor_io::vn_mnpd** = "mnpd"
- character(len= ∗), parameter **nestor_io::vn_xmpot** = "xmpot"
- character(len= ∗), parameter **nestor_io::vn_xnpot** = "xnpot"
- character(len= ∗), parameter **nestor_io::vn_potvac** = "potvac"
- character(len= ∗), parameter **nestor_io::vn_brv** = "brv"
- character(len= ∗), parameter **nestor_io::vn_bphiv** = "bphiv"
- character(len= ∗), parameter **nestor_io::vn_bzv** = "bzv"
- character(len= ∗), parameter **nestor_io::vn_bsubvvac** = "bsubvvac"
- character(len= ∗), parameter **nestor_io::vn_amatsav** = "amatsav"
- character(len= ∗), parameter **nestor_io::vn_bvecsav** = "bvecsav"
- character(len= ∗), parameter **nestor_io::vn_mnpd2** = "mnpd2"
- character(len= ∗), parameter **nestor_io::vn_r1b** = "r1b"
- character(len= ∗), parameter **nestor_io::vn_rub** = "rub"
- character(len= ∗), parameter **nestor_io::vn_rvb** = "rvb"
- character(len= ∗), parameter **nestor_io::vn_z1b** = "z1b"
- character(len= ∗), parameter **nestor_io::vn_zub** = "zub"
- character(len= ∗), parameter **nestor_io::vn_zvb** = "zvb"
- character(len= ∗), parameter **nestor_io::vn_ruu** = "ruu"
- character(len= ∗), parameter **nestor_io::vn_ruv** = "ruv"
- character(len= ∗), parameter **nestor_io::vn_rvv** = "rvv"
- character(len= ∗), parameter **nestor_io::vn_zuu** = "zuu"
- character(len= ∗), parameter **nestor_io::vn_zuv** = "zuv"
- character(len= ∗), parameter **nestor_io::vn_zvv** = "zvv"
- character(len= ∗), parameter **nestor_io::vn_guu_b** = "guu_b"
- character(len= ∗), parameter **nestor_io::vn_guv_b** = "guv_b"
- character(len= ∗), parameter **nestor_io::vn_gvv_b** = "gvv_b"

- character(len= ∗), parameter **nestor_io::vn_rzb2** = "rzb2"
- character(len= ∗), parameter **nestor_io::vn_snr** = "snr"
- character(len= ∗), parameter **nestor_io::vn_snv** = "snv"
- character(len= ∗), parameter **nestor_io::vn_snz** = "snz"
- character(len= ∗), parameter **nestor_io::vn_drv** = "drv"
- character(len= ∗), parameter **nestor_io::vn_auu** = "auu"
- character(len= ∗), parameter **nestor_io::vn_auv** = "auv"
- character(len= ∗), parameter **nestor_io::vn_avv** = "avv"
- character(len= ∗), parameter **nestor_io::vn_rcosuv** = "rcosuv"
- character(len= ∗), parameter **nestor_io::vn_rsinuv** = "rsinuv"
- character(len= ∗), parameter **nestor_io::vn_brad** = "brad"
- character(len= ∗), parameter **nestor_io::vn_bphi** = "bphi"
- character(len= ∗), parameter **nestor_io::vn_bz** = "bz"
- character(len= ∗), parameter **nestor_io::vn_bexu** = "bexu"
- character(len= ∗), parameter **nestor_io::vn_bexv** = "bexv"
- character(len= ∗), parameter **nestor_io::vn_bexn** = "bexn"
- character(len= ∗), parameter **nestor_io::vn_bexni** = "bexni"
- character(len= ∗), parameter **nestor_io::vn_grpmn** = "grpmn"
- character(len= ∗), parameter **nestor_io::vn_adp** = "adp"
- character(len= ∗), parameter **nestor_io::vn_adm** = "adm"
- character(len= ∗), parameter **nestor_io::vn_cma** = "cma"
- character(len= ∗), parameter **nestor_io::vn_sqrtc** = "sqrtc"
- character(len= ∗), parameter **nestor_io::vn_sqrta** = "sqrta"
- character(len= ∗), parameter **nestor_io::vn_delt1u** = "delt1u"
- character(len= ∗), parameter **nestor_io::vn_azp1u** = "azp1u"
- character(len= ∗), parameter **nestor_io::vn_azm1u** = "azm1u"
- character(len= ∗), parameter **nestor_io::vn_cma11u** = "cma11u"
- character(len= ∗), parameter **nestor_io::vn_r1p** = "r1p"
- character(len= ∗), parameter **nestor_io::vn_r1m** = "r1m"
- character(len= ∗), parameter **nestor_io::vn_r0p** = "r0p"
- character(len= ∗), parameter **nestor_io::vn_r0m** = "r0m"
- character(len= ∗), parameter **nestor_io::vn_ra1p** = "ra1p"
- character(len= ∗), parameter **nestor_io::vn_ra1m** = "ra1m"
- character(len= ∗), parameter **nestor_io::vn_sqad1u** = "sqad1u"
- character(len= ∗), parameter **nestor_io::vn_sqad2u** = "sqad2u"
- character(len= ∗), parameter **nestor_io::vn_all_tlp** = "all_tlp"
- character(len= ∗), parameter **nestor_io::vn_all_tlm** = "all_tlm"
- character(len= ∗), parameter **nestor_io::vn_all_slp** = "all_slp"
- character(len= ∗), parameter **nestor_io::vn_all_slm** = "all_slm"
- character(len= ∗), parameter **nestor_io::vn_m_map** = "m_map"
- character(len= ∗), parameter **nestor_io::vn_n_map** = "n_map"
- character(len= ∗), parameter **nestor_io::vn_green** = "green"
- character(len= ∗), parameter **nestor_io::vn_greenp** = "greenp"
- character(len= ∗), parameter **nestor_io::vn_tanu** = "tanu"
- character(len= ∗), parameter **nestor_io::vn_tanv** = "tanv"
- character(len= ∗), parameter **nestor_io::vn_gstore** = "gstore"
- character(len= ∗), parameter **nestor_io::vn_grpmn_m_map** = "grpmn_m_map"
- character(len= ∗), parameter **nestor_io::vn_grpmn_n_map** = "grpmn_n_map"
- character(len= ∗), parameter **nestor_io::vn_imirr** = "imirr"
- character(len= ∗), parameter **nestor_io::vn_amatrix** = "amatrix"
- character(len= ∗), parameter **nestor_io::vn_potu** = "potu"
- character(len= ∗), parameter **nestor_io::vn_potv** = "potv"
- character(len= ∗), parameter **nestor_io::vn_bsubu** = "bsubu"
- character(len= ∗), parameter **nestor_io::vn_bsubv** = "bsubv"

7.63.1 Detailed Description

Input and Output for stand-alone NESTOR.

7.64 src/NESTOR/data/vac_persistent.f90 File Reference

Variables

- integer, dimension(:), allocatable **vac_persistent::imirr**
- real(rprec), dimension(:), allocatable **vac_persistent::sinper**
- real(rprec), dimension(:), allocatable **vac_persistent::cosper**
- real(rprec), dimension(:), allocatable **vac_persistent::sinuv**
- real(rprec), dimension(:), allocatable **vac_persistent::cosuv**
- real(rprec), dimension(:), allocatable **vac_persistent::tanu**
- real(rprec), dimension(:), allocatable **vac_persistent::tanv**
- real(rprec), dimension(:), allocatable **vac_persistent::tanu_1d**
- real(rprec), dimension(:), allocatable **vac_persistent::tanv_1d**
- real(rprec), dimension(:), allocatable **vac_persistent::xmpot**
- real(rprec), dimension(:), allocatable **vac_persistent::xnpot**
- real(rprec), dimension(:), allocatable **vac_persistent::csign**
- real(rprec), dimension(:,:), allocatable **vac_persistent::sinu**
- real(rprec), dimension(:,:), allocatable **vac_persistent::cosu**
- real(rprec), dimension(:,:), allocatable **vac_persistent::sinv**
- real(rprec), dimension(:,:), allocatable **vac_persistent::cosv**
- real(rprec), dimension(:,:), allocatable **vac_persistent::sinui**
- real(rprec), dimension(:,:), allocatable **vac_persistent::cosui**
- real(rprec), dimension(:,:), allocatable **vac_persistent::sinu1**
- real(rprec), dimension(:,:), allocatable **vac_persistent::cosu1**
- real(rprec), dimension(:,:), allocatable **vac_persistent::sinv1**
- real(rprec), dimension(:,:), allocatable **vac_persistent::cosv1**
- real(rprec), dimension(:,:,:), allocatable **vac_persistent::cmns**
- real(rprec), dimension(:), allocatable **vac_persistent::bsubu_sur**
- real(rprec), dimension(:), allocatable **vac_persistent::bsubv_sur**
- real(rprec), dimension(:), allocatable **vac_persistent::bsupu_sur**
- real(rprec), dimension(:), allocatable **vac_persistent::bsupv_sur**

7.65 src/NESTOR/data/vacmod.f90 File Reference

Functions/Subroutines

- subroutine **vacmod::allocate_nestor**
- subroutine **vacmod::free_mem_nestor**

Variables

- real(rprec), parameter **vacmod::p5** = cp5
- real(rprec), parameter **vacmod::two** = c2p0
- real(rprec) **vacmod::bsubvvac**
- real(rprec) **vacmod::pi2**
- real(rprec) **vacmod::pi3**
- real(rprec) **vacmod::pi4**
- real(rprec) **vacmod::alp**
- real(rprec) **vacmod::alu**
- real(rprec) **vacmod::alv**
- real(rprec) **vacmod::alvp**
- real(rprec) **vacmod::onp**
- real(rprec) **vacmod::onp2**
- logical **vacmod::precal_done**
- real(rprec), dimension(:), allocatable, target **vacmod::potvac**
- real(rprec), dimension(:), allocatable **vacmod::m_map_wrt**
- real(rprec), dimension(:), allocatable **vacmod::n_map_wrt**
- real(rprec), dimension(:), allocatable **vacmod::bvecsav**
- real(rprec), dimension(:), allocatable **vacmod::amatsav**
- real(rprec), dimension(:), allocatable **vacmod::bexni**
- real(rprec), dimension(:), allocatable **vacmod::brv**
- real(rprec), dimension(:), allocatable **vacmod::bphiv**
- real(rprec), dimension(:), allocatable **vacmod::bzv**
- real(rprec), dimension(:), allocatable **vacmod::bsqvac**
- real(rprec), dimension(:), allocatable **vacmod::r1b**
- real(rprec), dimension(:), allocatable **vacmod::rub**
- real(rprec), dimension(:), allocatable **vacmod::rvb**
- real(rprec), dimension(:), allocatable **vacmod::z1b**
- real(rprec), dimension(:), allocatable **vacmod::zub**
- real(rprec), dimension(:), allocatable **vacmod::zvb**
- real(rprec), dimension(:), allocatable **vacmod::bexu**
- real(rprec), dimension(:), allocatable **vacmod::bexv**
- real(rprec), dimension(:), allocatable **vacmod::bexn**
- real(rprec), dimension(:), allocatable **vacmod::auu**
- real(rprec), dimension(:), allocatable **vacmod::auv**
- real(rprec), dimension(:), allocatable **vacmod::avv**
- real(rprec), dimension(:), allocatable **vacmod::snr**
- real(rprec), dimension(:), allocatable **vacmod::snv**
- real(rprec), dimension(:), allocatable **vacmod::snz**
- real(rprec), dimension(:), allocatable **vacmod::drv**
- real(rprec), dimension(:), allocatable **vacmod::guu_b**
- real(rprec), dimension(:), allocatable **vacmod::guv_b**
- real(rprec), dimension(:), allocatable **vacmod::gvv_b**
- real(rprec), dimension(:), allocatable **vacmod::rzb2**
- real(rprec), dimension(:), allocatable **vacmod::rcosuv**
- real(rprec), dimension(:), allocatable **vacmod::rsinuv**
- real(rprec), dimension(:), allocatable **vacmod::raxis_nestor**
- real(rprec), dimension(:), allocatable **vacmod::zaxis_nestor**
- real(rprec), dimension(:), allocatable **vacmod::bsubu**
- real(rprec), dimension(:), allocatable **vacmod::bsubv**
- real(rprec), dimension(:), allocatable **vacmod::potu**
- real(rprec), dimension(:), allocatable **vacmod::potv**
- real(rprec), dimension(:), allocatable **vacmod::amatrix**
- real(rprec), dimension(:), allocatable **vacmod::ruu**

- real(rprec), dimension(:), allocatable **vacmod::ruv**
- real(rprec), dimension(:), allocatable **vacmod::rvv**
- real(rprec), dimension(:), allocatable **vacmod::zuu**
- real(rprec), dimension(:), allocatable **vacmod::zuv**
- real(rprec), dimension(:), allocatable **vacmod::zvv**
- real(rprec), dimension(:), allocatable **vacmod::brad**
- real(rprec), dimension(:), allocatable **vacmod::bphi**
- real(rprec), dimension(:), allocatable **vacmod::bz**
- real(rprec), dimension(:,:), allocatable **vacmod::xpts**
- real(rprec), dimension(:), allocatable **vacmod::grpmn**
- real(rprec), dimension(:), allocatable **vacmod::grpmn_m_map_wrt**
- real(rprec), dimension(:), allocatable **vacmod::grpmn_n_map_wrt**
- real(rprec), dimension(:), allocatable **vacmod::gstore**
- real(rprec), dimension(:,:), allocatable **vacmod::green**
- real(rprec), dimension(:,:), allocatable **vacmod::greenp**
- real(rprec), dimension(:), allocatable **vacmod::r0p**
- real(rprec), dimension(:), allocatable **vacmod::r1p**
- real(rprec), dimension(:), allocatable **vacmod::r0m**
- real(rprec), dimension(:), allocatable **vacmod::r1m**
- real(rprec), dimension(:), allocatable **vacmod::sqrtc**
- real(rprec), dimension(:), allocatable **vacmod::sqrta**
- real(rprec), dimension(:), allocatable **vacmod::tlp2**
- real(rprec), dimension(:), allocatable **vacmod::tlp1**
- real(rprec), dimension(:), allocatable **vacmod::tlp**
- real(rprec), dimension(:), allocatable **vacmod::tlm2**
- real(rprec), dimension(:), allocatable **vacmod::tlm1**
- real(rprec), dimension(:), allocatable **vacmod::tlm**
- real(rprec), dimension(:), allocatable **vacmod::adp**
- real(rprec), dimension(:), allocatable **vacmod::adm**
- real(rprec), dimension(:), allocatable **vacmod::cma**
- real(rprec), dimension(:), allocatable **vacmod::ra1p**
- real(rprec), dimension(:), allocatable **vacmod::ra1m**
- real(rprec), dimension(:), allocatable **vacmod::slm**
- real(rprec), dimension(:), allocatable **vacmod::slp**
- real(rprec), dimension(:), allocatable **vacmod::tlpm**
- real(rprec), dimension(:), allocatable **vacmod::slpm**
- real(rprec), dimension(:), allocatable **vacmod::delt1u**
- real(rprec), dimension(:), allocatable **vacmod::azp1u**
- real(rprec), dimension(:), allocatable **vacmod::azm1u**
- real(rprec), dimension(:), allocatable **vacmod::cma11u**
- real(rprec), dimension(:), allocatable **vacmod::sqad1u**
- real(rprec), dimension(:), allocatable **vacmod::sqad2u**
- real(rprec), dimension(:,:), allocatable **vacmod::all_tlp**
- real(rprec), dimension(:,:), allocatable **vacmod::all_tlm**
- real(rprec), dimension(:,:), allocatable **vacmod::all_slp**
- real(rprec), dimension(:,:), allocatable **vacmod::all_slm**
- real(rprec), dimension(:), allocatable **vacmod::gsave**
- real(rprec), dimension(:), allocatable **vacmod::ga1**
- real(rprec), dimension(:), allocatable **vacmod::ga2**
- real(rprec), dimension(:), allocatable **vacmod::dsave**
- real(rprec), dimension(:,:,:), allocatable **vacmod::g1**
- real(rprec), dimension(:,:,:), allocatable **vacmod::g2**
- real(rprec), dimension(:,:,:), allocatable **vacmod::bcos**
- real(rprec), dimension(:,:,:), allocatable **vacmod::bsin**
- real(rprec), dimension(:,:,:), allocatable **vacmod::source**
- real(rprec), dimension(:,:,:,:), allocatable **vacmod::actemp**
- real(rprec), dimension(:,:,:,:), allocatable **vacmod::astemp**

7.66 src/NESTOR/data/vacmod0.f90 File Reference

Functions/Subroutines

- subroutine **vacmod0::set_nestor_sizes** (nfp, ntor, mpol, nzeta, ntheta, lasym)

Variables

- integer **vacmod0::mf**
- integer **vacmod0::nf**
- integer **vacmod0::nu**
- integer **vacmod0::nv**
- integer **vacmod0::mf1**
- integer **vacmod0::nf1**
- integer **vacmod0::mnpd**
- integer **vacmod0::mnpd2**
- integer **vacmod0::nuv**
- integer **vacmod0::nu2**
- integer **vacmod0::nu3**
- integer **vacmod0::nuv2**
- integer **vacmod0::nfper**
- integer **vacmod0::nvper**
- integer **vacmod0::nuv_tan**
- integer **vacmod0::nvp**
- integer **vacmod0::ndim**

7.67 src/NESTOR/fouri.f90 File Reference

Functions/Subroutines

- subroutine fouri (grpmn, gsource, amatrix, amatsq, bvec, wint, lasym)

7.67.1 Function/Subroutine Documentation

7.67.1.1 fouri()

```
subroutine fouri (
            real(rprec), dimension(mnpd,nv,nu3,ndim), intent(in) grpmn,
            real(rprec), dimension(nuv), intent(in) gsource,
            real(rprec), dimension(mnpd,mnpd,ndim**2), intent(out) amatrix,
            real(rprec), dimension(mnpd2,mnpd2), intent(out) amatsq,
            real(rprec), dimension(0:mf,-nf:nf,ndim), intent(inout) bvec,
            real(rprec), dimension(nuv2), intent(in) wint,
            logical, intent(in) lasym )
```

interior (int_ext=-1), exterior (int_ext=+1) neumann problem

Definition at line 2 of file fouri.f90.

7.68 src/NESTOR/fourp.f90 File Reference

Functions/Subroutines

- subroutine **fourp** (grpmn, grp)

7.69 src/NESTOR/greenf.f90 File Reference

Functions/Subroutines

- subroutine **greenf** (delgr, delgrp, ip)

7.70 src/NESTOR/nestor_main.f90 File Reference

Main program of stand-alone version of NESTOR.

Functions/Subroutines

- program nestor

  Main program of stand-alone version of NESTOR.

7.70.1 Detailed Description

Main program of stand-alone version of NESTOR.

7.71 src/NESTOR/precal.f90 File Reference

Functions/Subroutines

- subroutine **precal**

7.72 src/NESTOR/scalpot.f90 File Reference

Functions/Subroutines

- subroutine **scalpot** (bvec, amatrix, wint, ivacskip, lasym, m_map, n_map)

7.73 src/NESTOR/surface.f90 File Reference

Functions/Subroutines

- subroutine **surface** (rc, rs, zs, zc, xm, xn, mnmax, lasym, signgs)

7.74 src/NESTOR/vacuum.f90 File Reference

Functions/Subroutines

- subroutine **vacuum** (rmnc, rmns, zmns, zmnc, xm, xn, plascur, rbtor, wint, ivac_skip, ivac, mnmax, ier_flag, lasym, signgs, raxis, zaxis)

7.75 src/open_output_files.f90 File Reference

Open output files.

Functions/Subroutines

- subroutine open_output_files (extension, lfirst)

    Open output files.

7.75.1 Detailed Description

Open output files.

7.75.2 Function/Subroutine Documentation

7.75.2.1 open_output_files()

```
subroutine open_output_files (
            character(len=*) extension,
            logical lfirst )
```

Open output files.

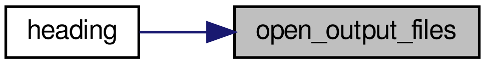
Parameters

extension	input file "extension": part after `'input.'` .
lfirst	flag to indicate if this is the first call to this routine or not

Definition at line 8 of file open_output_files.f90.

Referenced by heading().

Here is the caller graph for this function:



7.76 src/parse_extension.f File Reference

Parse the first command-line argument into a filename.

Functions/Subroutines

- subroutine parse_extension (file_to_parse, file_or_extension, lnc)
  Parse the first command-line argument into a filename.

7.76.1 Detailed Description

Parse the first command-line argument into a filename.

7.76.2 Function/Subroutine Documentation

7.76.2.1 parse_extension()

```
subroutine parse_extension (
             character(len=*), intent(inout) file_to_parse,
             character(len=*), intent(in) file_or_extension,
             logical, intent(out) lnc )
```

Parse the first command-line argument into a filename.

Parameters

file_to_parse	actual filename to read the input for VMEC from
file_or_extension	first command-line parameter given to VMEC
lnc	flag to indicate that a netCDF file is given

Definition at line 9 of file parse_extension.f.

7.77 src/precondn.f90 File Reference

Compute preconditioning matrix elements for R, Z force.

Functions/Subroutines

- subroutine precondn (lu1, bsq, gsqrt, r12, xs, xu12, xue, xuo, xodd, axm, axd, bxm, bxd, cx, eqfactor, trigmult)

  Compute preconditioning matrix elements for R, Z force.

7.77.1 Detailed Description

Compute preconditioning matrix elements for R, Z force.

7.77.2 Function/Subroutine Documentation

7.77.2.1 precondn()

```
subroutine precondn (
             real(rprec), dimension(nrzt), intent(in) lu1,
             real(rprec), dimension(nrzt), intent(in) bsq,
             real(rprec), dimension(nrzt), intent(in) gsqrt,
             real(rprec), dimension(nrzt), intent(in) r12,
             real(rprec), dimension(nrzt), intent(in) xs,
             real(rprec), dimension(nrzt), intent(in) xu12,
             real(rprec), dimension(nrzt), intent(in) xue,
             real(rprec), dimension(nrzt), intent(in) xuo,
             real(rprec), dimension(nrzt), intent(in) xodd,
             real(rprec), dimension(ns+1,2), intent(out) axm,
             real(rprec), dimension(ns+1,2), intent(out) axd,
             real(rprec), dimension(ns+1,2), intent(out) bxm,
             real(rprec), dimension(ns+1,2), intent(out) bxd,
             real(rprec), dimension(ns+1), intent(out) cx,
             real(rprec), dimension(ns), intent(out) eqfactor,
             real(rprec), dimension(nznt), intent(in) trigmult )
```

Compute preconditioning matrix elements for R, Z force.

Parameters

lu1	
bsq	
gsqrt	
r12	
xs	
xu12	
xue	
xuo	
xodd	

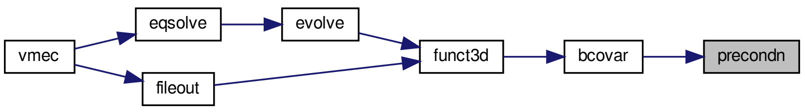
Parameters

axm	
axd	
bxm	
bxd	
cx	
eqfactor	
trigmult	

Definition at line 22 of file precondn.f90.

Referenced by bcovar().

Here is the caller graph for this function:



7.78 src/printout.f90 File Reference

Print iteration progress to screen and `threed1` output file.

Functions/Subroutines

- subroutine printout (i0, delt0, w0)

  Print iteration progress to screen and `threed1` output file.

7.78.1 Detailed Description

Print iteration progress to screen and `threed1` output file.

7.78.2 Function/Subroutine Documentation

7.78.2.1 printout()

```
subroutine printout (
             integer i0,
             real(rprec) delt0,
             real(rprec) w0 )
```

Print iteration progress to screen and `threed1` output file.

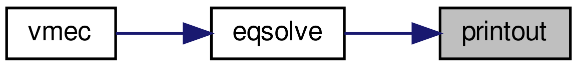
Parameters

i0	current iteration number
delt0	current time step
w0	current MHD energy

Definition at line 9 of file printout.f90.

Referenced by eqsolve().

Here is the caller graph for this function:



7.79 src/profil1d.f90 File Reference

Compute phip and iota profiles on full grid.

Functions/Subroutines

- subroutine profil1d (xc, xcdot, lreset)
  Compute phip and iota profiles on full grid.

7.79.1 Detailed Description

Compute phip and iota profiles on full grid.

7.79.2 Function/Subroutine Documentation

7.79.2.1 profil1d()

```
subroutine profil1d (
            real(rprec), dimension(neqs), intent(out) xc,
            real(rprec), dimension(neqs), intent(out) xcdot,
            logical, intent(in) lreset )
```

Compute phip and iota profiles on full grid.

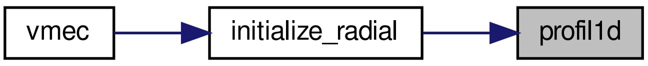
Parameters

xc	state vector of VMEC, i.e., all Fourier coefficients of R, Z and λ
xcdot	velocity vector in Fourier space
lreset	xc will be zeroes if this is true

Definition at line 9 of file profil1d.f90.

Referenced by initialize_radial().

Here is the caller graph for this function:



7.80 src/profil3d.f90 File Reference

Functions/Subroutines

- subroutine **profil3d** (rmn, zmn, lreset)

7.81 src/profile_functions.f File Reference

Functions/Subroutines

- real(rprec) function **pcurr** (xx)
- real(rprec) function **piota** (x)
- real(rprec) function **pmass** (xx)

7.82 src/read_indata.f90 File Reference

Functions/Subroutines

- subroutine **read_indata** (in_file, iunit, ier_flag)

7.83 src/read_wout_mod.f File Reference

Data Types

- interface read_wout_mod::read_wout_file

Functions/Subroutines

- subroutine **read_wout_mod::readw_and_open** (file_or_extension, ierr, iopen)
- subroutine **read_wout_mod::compute_currents** (ierror)
- subroutine **read_wout_mod::read_wout_deallocate**
- subroutine **read_wout_mod::tosuvspace** (s_in, u_in, v_in, gsqrt, bsupu, bsupv, jsupu, jsupv, lam)
- subroutine **read_wout_mod::loadrzl**

Variables

- character(len= ∗), parameter **read_wout_mod::vn_version** = 'version_'
- character(len= ∗), parameter **read_wout_mod::vn_extension** = 'input_extension'
- character(len= ∗), parameter **read_wout_mod::vn_mgrid** = 'mgrid_file'
- character(len= ∗), parameter **read_wout_mod::vn_magen** = 'wb'
- character(len= ∗), parameter **read_wout_mod::vn_therm** = 'wp'
- character(len= ∗), parameter **read_wout_mod::vn_gam** = 'gamma'
- character(len= ∗), parameter **read_wout_mod::vn_maxr** = 'rmax_surf'
- character(len= ∗), parameter **read_wout_mod::vn_minr** = 'rmin_surf'
- character(len= ∗), parameter **read_wout_mod::vn_maxz** = 'zmax_surf'
- character(len= ∗), parameter **read_wout_mod::vn_fp** = 'nfp'
- character(len= ∗), parameter **read_wout_mod::vn_radnod** = 'ns'
- character(len= ∗), parameter **read_wout_mod::vn_polmod** = 'mpol'
- character(len= ∗), parameter **read_wout_mod::vn_tormod** = 'ntor'
- character(len= ∗), parameter **read_wout_mod::vn_maxmod** = 'mnmax'
- character(len= ∗), parameter **read_wout_mod::vn_maxit** = 'niter'
- character(len= ∗), parameter **read_wout_mod::vn_actit** = 'itfsq'
- character(len= ∗), parameter **read_wout_mod::vn_asym** = 'lasym'
- character(len= ∗), parameter **read_wout_mod::vn_free** = 'lfreeb'
- character(len= ∗), parameter **read_wout_mod::vn_error** = 'ier_flag'
- character(len= ∗), parameter **read_wout_mod::vn_aspect** = 'aspect'
- character(len= ∗), parameter **read_wout_mod::vn_maxmod_nyq** = 'mnmax_nyq'
- character(len= ∗), parameter **read_wout_mod::vn_beta** = 'betatotal'
- character(len= ∗), parameter **read_wout_mod::vn_pbeta** = 'betapol'
- character(len= ∗), parameter **read_wout_mod::vn_tbeta** = 'betator'
- character(len= ∗), parameter **read_wout_mod::vn_abeta** = 'betaxis'
- character(len= ∗), parameter **read_wout_mod::vn_b0** = 'b0'
- character(len= ∗), parameter **read_wout_mod::vn_rbt0** = 'rbtor0'
- character(len= ∗), parameter **read_wout_mod::vn_rbt1** = 'rbtor'
- character(len= ∗), parameter **read_wout_mod::vn_sgs** = 'signgs'
- character(len= ∗), parameter **read_wout_mod::vn_lar** = 'IonLarmor'
- character(len= ∗), parameter **read_wout_mod::vn_modb** = 'volavgB'
- character(len= ∗), parameter **read_wout_mod::vn_ctor** = 'ctor'
- character(len= ∗), parameter **read_wout_mod::vn_amin** = 'Aminor_p'
- character(len= ∗), parameter **read_wout_mod::vn_rmaj** = 'Rmajor_p'
- character(len= ∗), parameter **read_wout_mod::vn_vol** = 'volume_p'
- character(len= ∗), parameter **read_wout_mod::vn_am** = 'am'
- character(len= ∗), parameter **read_wout_mod::vn_ai** = 'ai'
- character(len= ∗), parameter **read_wout_mod::vn_ac** = 'ac'
- character(len= ∗), parameter **read_wout_mod::vn_ah** = 'hot particle fraction'
- character(len= ∗), parameter **read_wout_mod::vn_atuname** = 'T-perp/T-par'
- character(len= ∗), parameter **read_wout_mod::vn_pmass_type** = 'pmass_type'
- character(len= ∗), parameter **read_wout_mod::vn_piota_type** = 'piota_type'
- character(len= ∗), parameter **read_wout_mod::vn_pcurr_type** = 'pcurr_type'
- character(len= ∗), parameter **read_wout_mod::vn_am_aux_s** = 'am_aux_s'

- character(len= ∗), parameter **read_wout_mod::vn_am_aux_f** = 'am_aux_f'
- character(len= ∗), parameter **read_wout_mod::vn_ai_aux_s** = 'ai_aux_s'
- character(len= ∗), parameter **read_wout_mod::vn_ai_aux_f** = 'ai_aux_f'
- character(len= ∗), parameter **read_wout_mod::vn_ac_aux_s** = 'ac_aux_s'
- character(len= ∗), parameter **read_wout_mod::vn_ac_aux_f** = 'ac_aux_f'
- character(len= ∗), parameter **read_wout_mod::vn_mse** = 'imse'
- character(len= ∗), parameter **read_wout_mod::vn_thom** = 'itse'
- character(len= ∗), parameter **read_wout_mod::vn_pmod** = 'xm'
- character(len= ∗), parameter **read_wout_mod::vn_tmod** = 'xn'
- character(len= ∗), parameter **read_wout_mod::vn_pmod_nyq** = 'xm_nyq'
- character(len= ∗), parameter **read_wout_mod::vn_tmod_nyq** = 'xn_nyq'
- character(len= ∗), parameter **read_wout_mod::vn_racc** = 'raxis_cc'
- character(len= ∗), parameter **read_wout_mod::vn_zacs** = 'zaxis_cs'
- character(len= ∗), parameter **read_wout_mod::vn_racs** = 'raxis_cs'
- character(len= ∗), parameter **read_wout_mod::vn_zacc** = 'zaxis_cc'
- character(len= ∗), parameter **read_wout_mod::vn_iotaf** = 'iotaf'
- character(len= ∗), parameter **read_wout_mod::vn_qfact** ='q-factor'
- character(len= ∗), parameter **read_wout_mod::vn_chi** ='chi'
- character(len= ∗), parameter **read_wout_mod::vn_chipf** ='chipf'
- character(len= ∗), parameter **read_wout_mod::vn_presf** = 'presf'
- character(len= ∗), parameter **read_wout_mod::vn_phi** = 'phi'
- character(len= ∗), parameter **read_wout_mod::vn_phipf** = 'phipf'
- character(len= ∗), parameter **read_wout_mod::vn_jcuru** = 'jcuru'
- character(len= ∗), parameter **read_wout_mod::vn_jcurv** = 'jcurv'
- character(len= ∗), parameter **read_wout_mod::vn_iotah** = 'iotas'
- character(len= ∗), parameter **read_wout_mod::vn_mass** = 'mass'
- character(len= ∗), parameter **read_wout_mod::vn_presh** = 'pres'
- character(len= ∗), parameter **read_wout_mod::vn_betah** = 'beta_vol'
- character(len= ∗), parameter **read_wout_mod::vn_buco** = 'buco'
- character(len= ∗), parameter **read_wout_mod::vn_bvco** = 'bvco'
- character(len= ∗), parameter **read_wout_mod::vn_vp** = 'vp'
- character(len= ∗), parameter **read_wout_mod::vn_specw** = 'specw'
- character(len= ∗), parameter **read_wout_mod::vn_phip** = 'phips'
- character(len= ∗), parameter **read_wout_mod::vn_jdotb** = 'jdotb'
- character(len= ∗), parameter **read_wout_mod::vn_overr** = 'over_r'
- character(len= ∗), parameter **read_wout_mod::vn_bgrv** = 'bdotgradv'
- character(len= ∗), parameter **read_wout_mod::vn_merc** = 'DMerc'
- character(len= ∗), parameter **read_wout_mod::vn_mshear** = 'DShear'
- character(len= ∗), parameter **read_wout_mod::vn_mwell** = 'DWell'
- character(len= ∗), parameter **read_wout_mod::vn_mcurr** = 'DCurr'
- character(len= ∗), parameter **read_wout_mod::vn_mgeo** = 'DGeod'
- character(len= ∗), parameter **read_wout_mod::vn_equif** = 'equif'
- character(len= ∗), parameter **read_wout_mod::vn_fsq** = 'fsqt'
- character(len= ∗), parameter **read_wout_mod::vn_wdot** = 'wdot'
- character(len= ∗), parameter **read_wout_mod::vn_ftolv** = 'ftolv'
- character(len= ∗), parameter **read_wout_mod::vn_fsql** = 'fsql'
- character(len= ∗), parameter **read_wout_mod::vn_fsqr** = 'fsqr'
- character(len= ∗), parameter **read_wout_mod::vn_fsqz** = 'fsqz'
- character(len= ∗), parameter **read_wout_mod::vn_extcur** = 'extcur'
- character(len= ∗), parameter **read_wout_mod::vn_curlab** = 'curlabel'
- character(len= ∗), parameter **read_wout_mod::vn_rmnc** = 'rmnc'
- character(len= ∗), parameter **read_wout_mod::vn_zmns** = 'zmns'
- character(len= ∗), parameter **read_wout_mod::vn_lmns** = 'lmns'
- character(len= ∗), parameter **read_wout_mod::vn_gmnc** = 'gmnc'
- character(len= ∗), parameter **read_wout_mod::vn_bmnc** = 'bmnc'

- character(len= ∗), parameter **read_wout_mod::vn_bsubumnc** = 'bsubumnc'
- character(len= ∗), parameter **read_wout_mod::vn_bsubvmnc** = 'bsubvmnc'
- character(len= ∗), parameter **read_wout_mod::vn_bsubsmns** = 'bsubsmns'
- character(len= ∗), parameter **read_wout_mod::vn_bsupumnc** = 'bsupumnc'
- character(len= ∗), parameter **read_wout_mod::vn_bsupvmnc** = 'bsupvmnc'
- character(len= ∗), parameter **read_wout_mod::vn_rmns** = 'rmns'
- character(len= ∗), parameter **read_wout_mod::vn_zmnc** = 'zmnc'
- character(len= ∗), parameter **read_wout_mod::vn_lmnc** = 'lmnc'
- character(len= ∗), parameter **read_wout_mod::vn_gmns** = 'gmns'
- character(len= ∗), parameter **read_wout_mod::vn_bmns** = 'bmns'
- character(len= ∗), parameter **read_wout_mod::vn_bsubumns** = 'bsubumns'
- character(len= ∗), parameter **read_wout_mod::vn_bsubvmns** = 'bsubvmns'
- character(len= ∗), parameter **read_wout_mod::vn_bsubsmnc** = 'bsubsmnc'
- character(len= ∗), parameter **read_wout_mod::vn_bsupumns** = 'bsupumns'
- character(len= ∗), parameter **read_wout_mod::vn_bsupvmns** = 'bsupvmns'
- character(len= ∗), parameter **read_wout_mod::vn_bsubumnc_sur** = 'bsubumnc_sur'
- character(len= ∗), parameter **read_wout_mod::vn_bsubvmnc_sur** = 'bsubvmnc_sur'
- character(len= ∗), parameter **read_wout_mod::vn_bsupumnc_sur** = 'bsupumnc_sur'
- character(len= ∗), parameter **read_wout_mod::vn_bsupvmnc_sur** = 'bsupvmnc_sur'
- character(len= ∗), parameter **read_wout_mod::vn_bsubumns_sur** = 'bsubumns_sur'
- character(len= ∗), parameter **read_wout_mod::vn_bsubvmns_sur** = 'bsubvmns_sur'
- character(len= ∗), parameter **read_wout_mod::vn_bsupumns_sur** = 'bsupumns_sur'
- character(len= ∗), parameter **read_wout_mod::vn_bsupvmns_sur** = 'bsupvmns_sur'
- character(len= ∗), parameter **read_wout_mod::vn_rbc** = 'rbc'
- character(len= ∗), parameter **read_wout_mod::vn_zbs** = 'zbs'
- character(len= ∗), parameter **read_wout_mod::vn_rbs** = 'rbs'
- character(len= ∗), parameter **read_wout_mod::vn_zbc** = 'zbc'
- character(len= ∗), parameter **read_wout_mod::vn_potvac** = 'potvac'
- character(len= ∗), parameter **read_wout_mod::ln_version** = 'VMEC Version'
- character(len= ∗), parameter **read_wout_mod::ln_extension** = 'Input file extension'
- character(len= ∗), parameter **read_wout_mod::ln_mgrid** = 'MGRID file'
- character(len= ∗), parameter **read_wout_mod::ln_magen** = 'Magnetic Energy'
- character(len= ∗), parameter **read_wout_mod::ln_therm** = 'Thermal Energy'
- character(len= ∗), parameter **read_wout_mod::ln_gam** = 'Gamma'
- character(len= ∗), parameter **read_wout_mod::ln_maxr** = 'Maximum R'
- character(len= ∗), parameter **read_wout_mod::ln_minr** = 'Minimum R'
- character(len= ∗), parameter **read_wout_mod::ln_maxz** = 'Maximum Z'
- character(len= ∗), parameter **read_wout_mod::ln_fp** = 'Field Periods'
- character(len= ∗), parameter **read_wout_mod::ln_radnod** = 'Radial nodes'
- character(len= ∗), parameter **read_wout_mod::ln_polmod** = 'Poloidal modes'
- character(len= ∗), parameter **read_wout_mod::ln_tormod** = 'Toroidal modes'
- character(len= ∗), parameter **read_wout_mod::ln_maxmod** = 'Fourier modes'
- character(len= ∗), parameter **read_wout_mod::ln_maxmod_nyq** = 'Fourier modes (Nyquist)'
- character(len= ∗), parameter **read_wout_mod::ln_maxit** = 'Max iterations'
- character(len= ∗), parameter **read_wout_mod::ln_actit** = 'Actual iterations'
- character(len= ∗), parameter **read_wout_mod::ln_asym** = 'Asymmetry'
- character(len= ∗), parameter **read_wout_mod::ln_recon** = 'Reconstruction'
- character(len= ∗), parameter **read_wout_mod::ln_free** = 'Free boundary'
- character(len= ∗), parameter **read_wout_mod::ln_error** = 'Error flag'
- character(len= ∗), parameter **read_wout_mod::ln_aspect** = 'Aspect ratio'
- character(len= ∗), parameter **read_wout_mod::ln_beta** = 'Total beta'
- character(len= ∗), parameter **read_wout_mod::ln_pbeta** = 'Poloidal beta'
- character(len= ∗), parameter **read_wout_mod::ln_tbeta** = 'Toroidal beta'
- character(len= ∗), parameter **read_wout_mod::ln_abeta** = 'Beta axis'
- character(len= ∗), parameter **read_wout_mod::ln_b0** = 'RB-t over R axis'

- character(len= $*$), parameter **read_wout_mod::ln_rbt0** = 'RB-t axis'
- character(len= $*$), parameter **read_wout_mod::ln_rbt1** = 'RB-t edge'
- character(len= $*$), parameter **read_wout_mod::ln_sgs** = 'Sign jacobian'
- character(len= $*$), parameter **read_wout_mod::ln_lar** = 'Ion Larmor radius'
- character(len= $*$), parameter **read_wout_mod::ln_modb** = 'avg mod B'
- character(len= $*$), parameter **read_wout_mod::ln_ctor** = 'Toroidal current'
- character(len= $*$), parameter **read_wout_mod::ln_amin** = 'minor radius'
- character(len= $*$), parameter **read_wout_mod::ln_rmaj** = 'major radius'
- character(len= $*$), parameter **read_wout_mod::ln_vol** = 'Plasma volume'
- character(len= $*$), parameter **read_wout_mod::ln_mse** = 'Number of MSE points'
- character(len= $*$), parameter **read_wout_mod::ln_thom** = 'Number of Thompson scattering points'
- character(len= $*$), parameter **read_wout_mod::ln_am** = 'Specification parameters for mass(s)'
- character(len= $*$), parameter **read_wout_mod::ln_ac** = 'Specification parameters for $<$J$>$(s)'
- character(len= $*$), parameter **read_wout_mod::ln_ai** = 'Specification parameters for iota(s)'
- character(len= $*$), parameter **read_wout_mod::ln_pmass_type** = 'Profile type specifier for mass(s)'
- character(len= $*$), parameter **read_wout_mod::ln_pcurr_type** = 'Profile type specifier for $<$J$>$(s)'
- character(len= $*$), parameter **read_wout_mod::ln_piota_type** = 'Profile type specifier for iota(s)'
- character(len= $*$), parameter **read_wout_mod::ln_am_aux_s** = 'Auxiliary-s parameters for mass(s)'
- character(len= $*$), parameter **read_wout_mod::ln_am_aux_f** = 'Auxiliary-f parameters for mass(s)'
- character(len= $*$), parameter **read_wout_mod::ln_ac_aux_s** = 'Auxiliary-s parameters for $<$J$>$(s)'
- character(len= $*$), parameter **read_wout_mod::ln_ac_aux_f** = 'Auxiliary-f parameters for $<$J$>$(s)'
- character(len= $*$), parameter **read_wout_mod::ln_ai_aux_s** = 'Auxiliary-s parameters for iota(s)'
- character(len= $*$), parameter **read_wout_mod::ln_ai_aux_f** = 'Auxiliary-f parameters for iota(s)'
- character(len= $*$), parameter **read_wout_mod::ln_pmod** = 'Poloidal mode numbers'
- character(len= $*$), parameter **read_wout_mod::ln_tmod** = 'Toroidal mode numbers'
- character(len= $*$), parameter **read_wout_mod::ln_pmod_nyq** = 'Poloidal mode numbers (Nyquist)'
- character(len= $*$), parameter **read_wout_mod::ln_tmod_nyq** = 'Toroidal mode numbers (Nyquist)'
- character(len= $*$), parameter **read_wout_mod::ln_racc** = 'raxis (cosnv)'
- character(len= $*$), parameter **read_wout_mod::ln_racs** = 'raxis (sinnv)'
- character(len= $*$), parameter **read_wout_mod::ln_zacs** = 'zaxis (sinnv)'
- character(len= $*$), parameter **read_wout_mod::ln_zacc** = 'zaxis (cosnv)'
- character(len= $*$), parameter **read_wout_mod::ln_iotaf** = 'iota on full mesh'
- character(len= $*$), parameter **read_wout_mod::ln_qfact** = 'q-factor on full mesh'
- character(len= $*$), parameter **read_wout_mod::ln_presf** = 'pressure on full mesh'
- character(len= $*$), parameter **read_wout_mod::ln_phi** = 'Toroidal flux on full mesh'
- character(len= $*$), parameter **read_wout_mod::ln_phipf** = 'd(phi)/ds: Toroidal flux deriv on full mesh'
- character(len= $*$), parameter **read_wout_mod::ln_chi** = 'Poloidal flux on full mesh'
- character(len= $*$), parameter **read_wout_mod::ln_chipf** = 'd(chi)/ds: Poroidal flux deriv on full mesh'
- character(len= $*$), parameter **read_wout_mod::ln_jcuru** = 'j dot gradu full'
- character(len= $*$), parameter **read_wout_mod::ln_jcurv** = 'j dot gradv full'
- character(len= $*$), parameter **read_wout_mod::ln_iotah** = 'iota half'
- character(len= $*$), parameter **read_wout_mod::ln_mass** = 'mass half'
- character(len= $*$), parameter **read_wout_mod::ln_presh** = 'pressure half'
- character(len= $*$), parameter **read_wout_mod::ln_betah** = 'beta half'
- character(len= $*$), parameter **read_wout_mod::ln_buco** = 'bsubu half'
- character(len= $*$), parameter **read_wout_mod::ln_bvco** = 'bsubv half'
- character(len= $*$), parameter **read_wout_mod::ln_vp** = 'volume deriv half'
- character(len= $*$), parameter **read_wout_mod::ln_specw** = 'Spectral width half'
- character(len= $*$), parameter **read_wout_mod::ln_phip** = 'tor flux deriv over 2pi half'
- character(len= $*$), parameter **read_wout_mod::ln_jdotb** = 'J dot B'
- character(len= $*$), parameter **read_wout_mod::ln_bgrv** = 'B dot grad v'
- character(len= $*$), parameter **read_wout_mod::ln_merc** = 'Mercier criterion'
- character(len= $*$), parameter **read_wout_mod::ln_mshear** = 'Shear Mercier'
- character(len= $*$), parameter **read_wout_mod::ln_mwell** = 'Well Mercier'
- character(len= $*$), parameter **read_wout_mod::ln_mcurr** = 'Current Mercier'

- character(len= ∗), parameter **read_wout_mod::ln_mgeo** = 'Geodesic Mercier'
- character(len= ∗), parameter **read_wout_mod::ln_equif** ='Average force balance'
- character(len= ∗), parameter **read_wout_mod::ln_fsq** = 'Residual decay'
- character(len= ∗), parameter **read_wout_mod::ln_wdot** = 'Wdot decay'
- character(len= ∗), parameter **read_wout_mod::ln_extcur** = 'External coil currents'
- character(len= ∗), parameter **read_wout_mod::ln_fsqr** = 'Residual decay - radial'
- character(len= ∗), parameter **read_wout_mod::ln_fsqz** = 'Residual decay - vertical'
- character(len= ∗), parameter **read_wout_mod::ln_fsql** = 'Residual decay - hoop'
- character(len= ∗), parameter **read_wout_mod::ln_ftolv** = 'Residual decay - requested'
- character(len= ∗), parameter **read_wout_mod::ln_curlab** = 'External current names'
- character(len= ∗), parameter **read_wout_mod::ln_rmnc** = 'cosmn component of cylindrical R, full mesh'
- character(len= ∗), parameter **read_wout_mod::ln_zmns** = 'sinmn component of cylindrical Z, full mesh'
- character(len= ∗), parameter **read_wout_mod::ln_lmns** = 'sinmn component of lambda, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_gmnc** = 'cosmn component of jacobian, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bmnc** = 'cosmn component of mod-B, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubumnc** = 'cosmn covariant u-component of B, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubvmnc** = 'cosmn covariant v-component of B, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubsmns** = 'sinmn covariant s-component of B, full mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubumnc_sur** = 'cosmn bsubu of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsubvmnc_sur** = 'cosmn bsubv of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsupumnc_sur** = 'cosmn bsupu of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsupvmnc_sur** = 'cosmn bsupv of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsupumnc** = 'BSUPUmnc half'
- character(len= ∗), parameter **read_wout_mod::ln_bsupvmnc** = 'BSUPVmnc half'
- character(len= ∗), parameter **read_wout_mod::ln_rmns** = 'sinmn component of cylindrical R, full mesh'
- character(len= ∗), parameter **read_wout_mod::ln_zmnc** = 'cosmn component of cylindrical Z, full mesh'
- character(len= ∗), parameter **read_wout_mod::ln_lmnc** = 'cosmn component of lambda, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_gmns** = 'sinmn component of jacobian, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bmns** = 'sinmn component of mod-B, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubumns** = 'sinmn covariant u-component of B, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubvmns** = 'sinmn covariant v-component of B, half mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubsmnc** = 'cosmn covariant s-component of B, full mesh'
- character(len= ∗), parameter **read_wout_mod::ln_bsubumns_sur** = 'sinmn bsubu of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsubvmns_sur** = 'sinmn bsubv of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsupumns_sur** = 'sinmn bsupu of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsupvmns_sur** = 'sinmn bsupv of B, surface'
- character(len= ∗), parameter **read_wout_mod::ln_bsupumns** = 'BSUPUmns half'
- character(len= ∗), parameter **read_wout_mod::ln_bsupvmns** = 'BSUPVmns half'
- character(len= ∗), parameter **read_wout_mod::ln_rbc** = 'Initial boundary R cos(mu-nv) coefficients'
- character(len= ∗), parameter **read_wout_mod::ln_zbs** = 'Initial boundary Z sin(mu-nv) coefficients'
- character(len= ∗), parameter **read_wout_mod::ln_rbs** = 'Initial boundary R sin(mu-nv) coefficients'
- character(len= ∗), parameter **read_wout_mod::ln_zbc** = 'Initial boundary Z cos(mu-nv) coefficients'
- character(len= ∗), parameter **read_wout_mod::ln_potvac** = 'Vacuum Potential on Boundary'
- integer **read_wout_mod::nfp**
- integer **read_wout_mod::ns**
- integer **read_wout_mod::mpol**
- integer **read_wout_mod::ntor**
- integer **read_wout_mod::mnmax**
- integer **read_wout_mod::mnmax_nyq**

- integer **read_wout_mod::itfsq**
- integer **read_wout_mod::niter**
- integer **read_wout_mod::iasym**
- integer **read_wout_mod::ierr_vmec**
- integer **read_wout_mod::imse**
- integer **read_wout_mod::itse**
- integer **read_wout_mod::nstore_seq**
- integer **read_wout_mod::isnodes**
- integer **read_wout_mod::ipnodes**
- integer **read_wout_mod::imatch_phiedge**
- integer **read_wout_mod::isigng**
- integer **read_wout_mod::mnyq**
- integer **read_wout_mod::nnyq**
- integer **read_wout_mod::ntmax**
- real(rprec) **read_wout_mod::wb**
- real(rprec) **read_wout_mod::wp**
- real(rprec) **read_wout_mod::gamma**
- real(rprec) **read_wout_mod::pfac**
- real(rprec) **read_wout_mod::rmax_surf**
- real(rprec) **read_wout_mod::rmin_surf**
- real(rprec) **read_wout_mod::zmax_surf**
- real(rprec) **read_wout_mod::aspect**
- real(rprec) **read_wout_mod::betatot**
- real(rprec) **read_wout_mod::betapol**
- real(rprec) **read_wout_mod::betator**
- real(rprec) **read_wout_mod::betaxis**
- real(rprec) **read_wout_mod::b0**
- real(rprec) **read_wout_mod::tswgt**
- real(rprec) **read_wout_mod::msewgt**
- real(rprec) **read_wout_mod::flmwgt**
- real(rprec) **read_wout_mod::bcwgt**
- real(rprec) **read_wout_mod::phidiam**
- real(rprec) **read_wout_mod::version_**
- real(rprec) **read_wout_mod::delphid**
- real(rprec) **read_wout_mod::ionlarmor**
- real(rprec) **read_wout_mod::volavgb**
- real(rprec) **read_wout_mod::fsql**
- real(rprec) **read_wout_mod::fsqr**
- real(rprec) **read_wout_mod::fsqz**
- real(rprec) **read_wout_mod::ftolv**
- real(rprec) **read_wout_mod::aminor**
- real(rprec) **read_wout_mod::rmajor**
- real(rprec) **read_wout_mod::volume**
- real(rprec) **read_wout_mod::rbtor**
- real(rprec) **read_wout_mod::rbtor0**
- real(rprec) **read_wout_mod::itor**
- real(rprec) **read_wout_mod::machsq**
- real(rprec), dimension(:,:,:,:), allocatable **read_wout_mod::rzl_local**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::rmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::zmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::lmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::rmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::zmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::lmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bmnc**

- real(rprec), dimension(:,:), allocatable **read_wout_mod::gmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsubumnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsubvmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsubsmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsupumnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsupvmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::currvmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::currumnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bbc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::raxis**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::zaxis**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::gmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsubumns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsubvmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsubsmnc**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsupumns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::bsupvmns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::currumns**
- real(rprec), dimension(:,:), allocatable **read_wout_mod::currvmns**
- real(rprec), dimension(:), allocatable **read_wout_mod::iotas**
- real(rprec), dimension(:), allocatable **read_wout_mod::iotaf**
- real(rprec), dimension(:), allocatable **read_wout_mod::presf**
- real(rprec), dimension(:), allocatable **read_wout_mod::phipf**
- real(rprec), dimension(:), allocatable **read_wout_mod::mass**
- real(rprec), dimension(:), allocatable **read_wout_mod::pres**
- real(rprec), dimension(:), allocatable **read_wout_mod::beta_vol**
- real(rprec), dimension(:), allocatable **read_wout_mod::xm**
- real(rprec), dimension(:), allocatable **read_wout_mod::xn**
- real(rprec), dimension(:), allocatable **read_wout_mod::qfact**
- real(rprec), dimension(:), allocatable **read_wout_mod::chipf**
- real(rprec), dimension(:), allocatable **read_wout_mod::phi**
- real(rprec), dimension(:), allocatable **read_wout_mod::chi**
- real(rprec), dimension(:), allocatable **read_wout_mod::xm_nyq**
- real(rprec), dimension(:), allocatable **read_wout_mod::xn_nyq**
- real(rprec), dimension(:), allocatable **read_wout_mod::phip**
- real(rprec), dimension(:), allocatable **read_wout_mod::buco**
- real(rprec), dimension(:), allocatable **read_wout_mod::bvco**
- real(rprec), dimension(:), allocatable **read_wout_mod::vp**
- real(rprec), dimension(:), allocatable **read_wout_mod::overr**
- real(rprec), dimension(:), allocatable **read_wout_mod::jcuru**
- real(rprec), dimension(:), allocatable **read_wout_mod::jcurv**
- real(rprec), dimension(:), allocatable **read_wout_mod::specw**
- real(rprec), dimension(:), allocatable **read_wout_mod::jdotb**
- real(rprec), dimension(:), allocatable **read_wout_mod::bdotgradv**
- real(rprec), dimension(:), allocatable **read_wout_mod::fsqt**
- real(rprec), dimension(:), allocatable **read_wout_mod::wdot**
- real(rprec), dimension(:), allocatable **read_wout_mod::am**
- real(rprec), dimension(:), allocatable **read_wout_mod::ac**
- real(rprec), dimension(:), allocatable **read_wout_mod::ai**
- real(rprec), dimension(:), allocatable **read_wout_mod::am_aux_s**
- real(rprec), dimension(:), allocatable **read_wout_mod::am_aux_f**
- real(rprec), dimension(:), allocatable **read_wout_mod::ac_aux_s**
- real(rprec), dimension(:), allocatable **read_wout_mod::ac_aux_f**
- real(rprec), dimension(:), allocatable **read_wout_mod::ai_aux_s**

- real(rprec), dimension(:), allocatable **read_wout_mod::ai_aux_f**
- real(rprec), dimension(:), allocatable **read_wout_mod::dmerc**
- real(rprec), dimension(:), allocatable **read_wout_mod::dshear**
- real(rprec), dimension(:), allocatable **read_wout_mod::dwell**
- real(rprec), dimension(:), allocatable **read_wout_mod::dcurr**
- real(rprec), dimension(:), allocatable **read_wout_mod::dgeod**
- real(rprec), dimension(:), allocatable **read_wout_mod::equif**
- real(rprec), dimension(:), allocatable **read_wout_mod::extcur**
- real(rprec), dimension(:), allocatable **read_wout_mod::sknots**
- real(rprec), dimension(:), allocatable **read_wout_mod::ystark**
- real(rprec), dimension(:), allocatable **read_wout_mod::y2stark**
- real(rprec), dimension(:), allocatable **read_wout_mod::pknots**
- real(rprec), dimension(:), allocatable **read_wout_mod::ythom**
- real(rprec), dimension(:), allocatable **read_wout_mod::y2thom**
- real(rprec), dimension(:), allocatable **read_wout_mod::anglemse**
- real(rprec), dimension(:), allocatable **read_wout_mod::rmid**
- real(rprec), dimension(:), allocatable **read_wout_mod::qmid**
- real(rprec), dimension(:), allocatable **read_wout_mod::shear**
- real(rprec), dimension(:), allocatable **read_wout_mod::presmid**
- real(rprec), dimension(:), allocatable **read_wout_mod::alfa**
- real(rprec), dimension(:), allocatable **read_wout_mod::curmid**
- real(rprec), dimension(:), allocatable **read_wout_mod::rstark**
- real(rprec), dimension(:), allocatable **read_wout_mod::qmeas**
- real(rprec), dimension(:), allocatable **read_wout_mod::datastark**
- real(rprec), dimension(:), allocatable **read_wout_mod::rthom**
- real(rprec), dimension(:), allocatable **read_wout_mod::datathom**
- real(rprec), dimension(:), allocatable **read_wout_mod::dsiobt**
- real(rprec), dimension(:), allocatable **read_wout_mod::potvac**
- logical **read_wout_mod::lasym**
- logical **read_wout_mod::lthreed**
- logical **read_wout_mod::lwout_opened** =.false.
- character **read_wout_mod::mgrid_file**
- character **read_wout_mod::input_extension**
- character **read_wout_mod::pmass_type**
- character **read_wout_mod::pcurr_type**
- character **read_wout_mod::piota_type**

7.84 src/readin.f90 File Reference

Functions/Subroutines

- subroutine **readin** (input_file, ier_flag)

7.85 src/reset_params.f90 File Reference

Functions/Subroutines

- subroutine reset_params

7.85.1 Function/Subroutine Documentation

7.85.1.1 reset_params()

```
subroutine reset_params
```

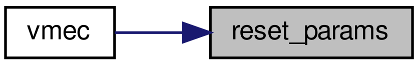
m=1 constraint (=t: apply correct, polar constraint; =f, apply approx. constraint)

Assume scaled mode; read in from mgrid in free-bdy mode

Definition at line 2 of file reset_params.f90.

Referenced by vmec().

Here is the caller graph for this function:



7.86 src/residue.f90 File Reference

Functions/Subroutines

- subroutine **residue** (gcr, gcz, gcl)
- subroutine **constrain_m1** (gcr, gcz)
- subroutine **scale_m1** (gcr, gcz)

7.87 src/restart_iter.f90 File Reference

Functions/Subroutines

- subroutine **restart_iter** (time_step)

7.88 src/safe_open_mod.f File Reference

Functions/Subroutines

- subroutine **safe_open_mod::safe_open** (iunit, istat, filename, filestat, fileform, record_in, access_in, delim_in)

7.89 src/scalfor.f90 File Reference

Functions/Subroutines

- subroutine **scalfor** (gcx, axm, bxm, axd, bxd, cx, iflag)

7.90 src/solver.f90 File Reference

Functions/Subroutines

- subroutine **solver** (amat, b, m, nrhs, info)

7.91 src/spectrum.f90 File Reference

Functions/Subroutines

- subroutine **spectrum** (rmn, zmn)

7.92 src/spline_akima.f File Reference

Functions/Subroutines

- subroutine **spline_akima** (x, y, xx, yy, npts, iflag)

7.93 src/spline_akima_int.f File Reference

Functions/Subroutines

- subroutine **spline_akima_int** (x, y, xx, yy, npts, iflag)

7.94 src/spline_cubic.f File Reference

Functions/Subroutines

- subroutine **spline_cubic** (x, y, xx, yy, n, iflag)
- subroutine **spline_nr** (x, y, n, yp1, ypn, y2)
- subroutine **splint_nr** (xa, ya, y2a, n, x, y)

7.95 src/spline_cubic_int.f File Reference

Functions/Subroutines

- subroutine **spline_cubic_int** (x, y, xx, yy, n, iflag)
- subroutine **spline_int** (x, y, n, yp1, ypn, y2)
- subroutine **splint_int** (xa, ya, y2a, n, x, y)

7.96 src/symforce.f90 File Reference

Functions/Subroutines

- subroutine **symforce** (ars, brs, crs, azs, bzs, czs, bls, cls, rcs, zcs, ara, bra, cra, aza, bza, cza, bla, cla, rca, zca)
- subroutine **symoutput** (bsq, gsqrt, bsubu, bsubv, bsupu, bsupv, bsubs, bsqa, gsqrta, bsubua, bsubva, bsupua, bsupva, bsubsa)

7.97 src/symrzl.f90 File Reference

Functions/Subroutines

- subroutine **symrzl** (r1s, rus, rvs, z1s, zus, zvs, lus, lvs, rcons, zcons, r1a, rua, rva, z1a, zua, zva, lua, lva, rcona, zcona)

7.98 src/tolower.f90 File Reference

Functions/Subroutines

- subroutine **tolower** (string)

7.99 src/tomnsp.f90 File Reference

Functions/Subroutines

- subroutine **tomnsps** (frzl_array, armn, brmn, crmn, azmn, bzmn, czmn, blmn, clmn, arcon, azcon)
- subroutine **tomnspa** (frzl_array, armn, brmn, crmn, azmn, bzmn, czmn, blmn, clmn, arcon, azcon)

7.100 src/totzsp.f90 File Reference

Functions/Subroutines

- subroutine totzsps (rzl_array, r11, ru1, rv1, z11, zu1, zv1, lu1, lv1, rcn1, zcn1)
- subroutine **convert_sym** (rmnss, zmncs)
- subroutine **totzspa** (rzl_array, r11, ru1, rv1, z11, zu1, zv1, lu1, lv1, rcn1, zcn1)
- subroutine **convert_asym** (rmnsc, zmncc)

7.100.1 Function/Subroutine Documentation

7.100.1.1 totzsps()

```
subroutine totzsps (
            real(rprec), dimension(ns,0:ntor,0:mpol1,3*ntmax), intent(inout), target rzl_↩
array,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) r11,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) ru1,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) rv1,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) z11,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) zu1,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) zv1,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) lu1,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) lv1,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) rcn1,
            real(rprec), dimension(ns*nzeta*ntheta3,0:1), intent(out) zcn1 )
```

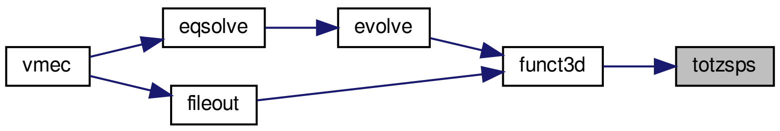
Parameters

out	*r11*	R
out	*ru1*	dR/dTheta
out	*rv1*	dR/dZeta
out	*z11*	Z
out	*zu1*	dZ/dTheta
out	*zv1*	dZ/dZeta
out	*lu1*	dLambda/dTheta
out	*lv1*	-dLambda/dZeta
out	*rcn1*	TODO: what is this?
out	*zcn1*	TODO: what is this?

Definition at line 2 of file totzsp.f90.

Referenced by funct3d().

Here is the caller graph for this function:



7.101 src/tridslv.f90 File Reference

Functions/Subroutines

- subroutine **tridslv** (a, d, b, c, jmin, jmax, mnd1, ns, nrhs)

7.102 src/vmec.f90 File Reference

Main program of VMEC.

Functions/Subroutines

- program vmec

  Main program of VMEC.

7.102.1 Detailed Description

Main program of VMEC.

7.103 src/wrout.f90 File Reference

Functions/Subroutines

- subroutine **wrout** (bsq, gsqrt, bsubu, bsubv, bsubs, bsupv, bsupu, rzl_array, gc_array, ier_flag)

Index