

Package: admvn (via r-universe)

October 16, 2026

Type Package

Title Multivariate Normal CDF and SUN Density with CppAD Derivatives

Version 0.2.0

Description Evaluates the cumulative distribution function of the multivariate normal distribution using specialized bivariate/trivariate algorithms for dimension ≤ 3 and Genz quasi-Monte Carlo for higher dimensions, with gradient and Hessian with respect to the upper integration limits computed via CppAD automatic differentiation. Also provides the SUN(2,2), SUN(3,3) and SUN(4,4) densities with gradient and Hessian with respect to a block-Cholesky parameter vector, Gauss-Hermite utilities, and KL minimisation of SUN approximations on weighted quadrature.

License GPL (≥ 2)

Depends R ($\geq 4.1.0$)

Imports Rcpp ($\geq 1.0.3$), RCppAD, stats, utils, statmod

LinkingTo Rcpp, RCppAD, RcppEigen, trustOptim

Suggests mvtnorm, mnormt, numDeriv, sn, trustOptim, testthat ($\geq 3.0.0$), knitr, rmarkdown, optimParallel

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

SystemRequirements OpenMP

Roxygen list(markdown = TRUE)

NeedsCompilation yes

Collate '000.R' 'admvn-package.R' 'RcppExports.R' 'pack_genz_ch.R' 'pmvn.R' 'pmvn_fun.R' 'make_sun_params.R' 'make_sun_hs_params.R' 'make_sun22_params.R' 'make_sun22_hs_params.R' 'make_sun32_hs_params.R' 'make_sun42_hs_params.R' 'make_sun52_hs_params.R' 'make_sun43_hs_params.R' 'make_sun44_params.R'

```
'make_sun44_hs_params.R' 'dsun.R' 'dsun_hs.R' 'dsun22.R'
'dsun22_hs.R' 'dsun32_hs.R' 'dsun42_hs.R' 'dsun52_hs.R'
'dsun43_hs.R' 'dsun44.R' 'dsun44_hs.R' 'gh_points.R'
'sun_bounds.R' 'sun_starts.R' 'sun_kl.R'
```

Config/roxygen2/version 8.1.0

Repository <https://eborgnine.r-universe.dev>

Date/Publication 2026-10-16 21:08:16 UTC

RemoteUrl <https://github.com/eborgnine/adlaplace>

RemoteRef main

RemoteSha 402cb9c50ba20f12ba8439d11cb9812574783497

RemoteSubdir admvn

Contents

admvn-package	4
dsun	4
dsun_fun	5
dsun_hs	6
dsun_hs_fun	7
dsun22	8
dsun22_fun	9
dsun22_hs	10
dsun22_hs_fun	10
dsun32_hs	11
dsun32_hs_fun	12
dsun42_hs	13
dsun42_hs_fun	13
dsun43_hs	14
dsun43_hs_fun	15
dsun44	16
dsun44_fun	17
dsun44_hs	18
dsun44_hs_fun	18
dsun52_hs	19
dsun52_hs_fun	20
evaluate_sun22_slice	21
evaluate_sun32_hs_slice	21
evaluate_sun33_hs_slice	22
evaluate_sun33_slice	23
evaluate_sun42_hs_slice	23
evaluate_sun43_hs_slice	24
evaluate_sun44_slice	24
evaluate_sun52_hs_slice	25
fit_sun22_quad	25
fit_sun32_hs_quad	26
fit_sun33_hs_quad	27

fit_sun33_quad	28
fit_sun42_hs_quad	29
fit_sun43_hs_quad	29
fit_sun44_quad	30
fit_sun52_hs_quad	31
gh_points	32
make_sun_hs_params	32
make_sun_params	33
make_sun22_hs_params	35
make_sun22_hs_start_from_gh	35
make_sun22_hs_start_from_normal	36
make_sun22_params	36
make_sun22_start_from_gh	37
make_sun22_start_from_normal	37
make_sun32_hs_params	38
make_sun32_hs_start_from_gh	38
make_sun32_hs_start_from_normal	39
make_sun33_hs_start_from_gh	39
make_sun33_hs_start_from_normal	40
make_sun33_start_from_gh	40
make_sun33_start_from_normal	41
make_sun42_hs_params	41
make_sun42_hs_start_from_gh	42
make_sun42_hs_start_from_normal	42
make_sun43_hs_params	43
make_sun43_hs_start_from_gh	44
make_sun43_hs_start_from_normal	44
make_sun44_hs_params	45
make_sun44_hs_start_from_gh	45
make_sun44_hs_start_from_normal	46
make_sun44_params	46
make_sun44_start_from_gh	47
make_sun44_start_from_normal	47
make_sun52_hs_params	48
make_sun52_hs_start_from_gh	48
make_sun52_hs_start_from_normal	49
pack_genz_ch	49
pmvn	50
pmvn_fun	51
print_sun_kl_fit	52
sun_mle	52
sun22_bounds	53
sun22_hs_bounds	53
sun32_hs_bounds	54
sun33_bounds	54
sun33_hs_bounds	55
sun42_hs_bounds	55
sun43_hs_bounds	56

sun44_bounds	56
sun44_hs_bounds	57
sun52_hs_bounds	57
weighted_cov	57
weighted_mean	58

Index 59

admvn-package	<i>Multivariate normal CDF and SUN density with CppAD derivatives</i>
---------------	---

Description

Genz quasi-Monte Carlo MVN probabilities with gradient and Hessian, SUN(3,3)/SUN(4,4) log-densities with derivatives with respect to a block-Cholesky parameter vector, and KL minimisation of SUN approximations on weighted Gauss-Hermite quadrature.

Author(s)

Maintainer: Patrick Brown <patrick.brown@gmail.com> [copyright holder]

Authors:

- Patrick Brown <patrick.brown@gmail.com> [copyright holder]

dsun	<i>SUN(3,3) density with derivatives</i>
------	--

Description

Evaluates the log-density (or density) of a SUN(3,3) distribution using the block-Cholesky parameterization from `make_sun_params()`. For a matrix of observations the returned value is the summed log-likelihood, or a weighted sum if `weights` is supplied. Derivatives are with respect to the 21-component parameter vector only; the data are treated as fixed.

Usage

```
dsun(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric vector of length 3 or an $n \times 3$ matrix of observations.
<code>par</code>	Numeric vector of length 21; see make_sun_params() .
<code>log</code>	If TRUE (default), return the log-density / log-likelihood.
<code>deriv</code>	Integer derivative order: 0 (default) value only, 1 also gradient, 2 also Hessian, all with respect to <code>par</code> .
<code>n_points, n_shifts, seed</code>	Quasi-Monte Carlo settings for the orthant probabilities in the SUN density.
<code>n_threads</code>	Number of OpenMP threads for parallel shard evaluation. Defaults to 1. Use 0 in <code>\$eval()</code> to reuse the value stored when the tape was built.
<code>weights</code>	Optional numeric vector of length <code>nrow(x)</code> . If supplied, returns $\sum_i w_i \log q(x_i)$ (and matching derivatives) instead of the unweighted sum. Defaults to equal weights of 1.

Value

A list with value and error. If `deriv >= 1`, also gradient (length 21). If `deriv >= 2`, also hessian (21×21).

See Also

[make_sun_params\(\)](#), [dsun_fun\(\)](#)

dsun_fun

Create a reusable SUN(3,3) log-likelihood tape

Description

Builds per-observation CppAD shard tapes for the summed SUN log-likelihood with the data `x` fixed and the independent variables equal to the 21-parameter vector. Use this for MLE: build once, then call `$eval(par)` at new parameter values. Pass `weights` for a weighted sum (e.g. Gauss–Hermite KL objectives).

Usage

```
dsun_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric vector of length 3 or $n \times 3$ matrix of observations fixed in the tape.
<code>par_seed</code>	Seed parameter vector used when building the tape.
<code>n_points, n_shifts, seed</code>	QMC settings passed to <code>dsun()</code> .
<code>n_threads</code>	Default number of OpenMP threads for <code>\$eval()</code> . When NULL, uses the maximum available OpenMP thread count.
<code>weights</code>	Optional numeric vector of length <code>nrow(x)</code> for a weighted log-likelihood $\sum_i w_i \log q(x_i)$.

Value

An object of class "admvn_sun_tape" with components `eval(par, log = TRUE, deriv = 0, n_threads = 0)` and `optim_fns()` for `optim()`: fn/gr share one `deriv = 1` evaluation cached by parameter value.

<code>dsun_hs</code>	<i>SUN(3,3) density with joint hyperspherical parameterization</i>
----------------------	--

Description

Like `dsun()`, but the length-21 parameter vector uses the joint hyperspherical map of `make_sun_hs_params()` instead of the block-Cholesky layout of `make_sun_params()`.

Usage

```
dsun_hs(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric vector of length 3 or an $n \times 3$ matrix of observations.
<code>par</code>	Numeric vector of length 21; see <code>make_sun_hs_params()</code> .
<code>log</code>	If TRUE (default), return the log-density / log-likelihood.
<code>deriv</code>	Integer derivative order: 0 (default) value only, 1 also gradient, 2 also Hessian, all with respect to <code>par</code> .

n_points, n_shifts, seed	Quasi-Monte Carlo settings for the orthant probabilities in the SUN density.
n_threads	Number of OpenMP threads for parallel shard evaluation. Defaults to 1. Use 0 in <code>\$eval()</code> to reuse the value stored when the tape was built.
weights	Optional numeric vector of length <code>nrow(x)</code> . If supplied, returns $\sum_i w_i \log q(x_i)$ (and matching derivatives) instead of the unweighted sum. Defaults to equal weights of 1.

Value

A list with value and error. If `deriv >= 1`, also gradient. If `deriv >= 2`, also hessian.

See Also

[make_sun_hs_params\(\)](#), [dsun_hs_fun\(\)](#), [dsun\(\)](#)

dsun_hs_fun	<i>Create a reusable SUN(3,3) hyperspherical log-likelihood tape</i>
-------------	--

Description

Like [dsun_fun\(\)](#), for the joint hyperspherical parameterization of [make_sun_hs_params\(\)](#).

Usage

```
dsun_hs_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

x	Numeric vector of length 3 or $n \times 3$ matrix of observations fixed in the tape.
par_seed	Seed parameter vector (length 21) for make_sun_hs_params() .
n_points, n_shifts, seed	QMC settings passed to dsun() .
n_threads	Default number of OpenMP threads for <code>\$eval()</code> . When NULL, uses the maximum available OpenMP thread count.
weights	Optional numeric vector of length <code>nrow(x)</code> for a weighted log-likelihood $\sum_i w_i \log q(x_i)$.

Value

An object of class "admvn_sun_tape" with `eval()` and `optim_fns()` (same interface as `dsun_fun()`).

See Also

`dsun_hs()`, `sun_mle()`

dsun22	<i>SUN(2,2) density with derivatives</i>
--------	--

Description

Evaluates the log-density (or density) of a SUN(2,2) distribution using the block-Cholesky parameterization from `make_sun22_params()`. For a matrix of observations the returned value is the summed log-likelihood, or a weighted sum if `weights` is supplied. Derivatives are with respect to the 10-component parameter vector only.

Usage

```
dsun22(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric vector of length 2 or an $n \times 2$ matrix of observations.
<code>par</code>	Numeric vector of length 10; see <code>make_sun22_params()</code> .
<code>log</code>	If TRUE (default), return the log-density / log-likelihood.
<code>deriv</code>	Integer derivative order: 0 (default) value only, 1 also gradient, 2 also Hessian, all with respect to <code>par</code> .
<code>n_points</code> , <code>n_shifts</code> , <code>seed</code>	Quasi-Monte Carlo settings for orthant CDFs.
<code>n_threads</code>	Number of OpenMP threads for parallel shard evaluation.
<code>weights</code>	Optional numeric vector of length <code>nrow(x)</code> .

Value

A list with value and error. If `deriv >= 1`, also gradient (length 10). If `deriv >= 2`, also hessian (10×10).

See Also

[make_sun22_params\(\)](#), [dsun22_fun\(\)](#), [dsun\(\)](#)

dsun22_fun

Create a reusable SUN(2,2) log-likelihood tape

Description

Builds per-observation CppAD shard tapes for the (weighted) SUN(2,2) log-likelihood with data `x` fixed. Use for KL / MLE: build once, then call `$eval(par)` or `$optim_fns()`.

Usage

```
dsun22_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric vector of length 2 or an $n \times 2$ matrix of observations.
<code>par_seed</code>	Seed parameter vector used when building the tape.
<code>n_points, n_shifts, seed</code>	Quasi-Monte Carlo settings for orthant CDFs.
<code>n_threads</code>	Default OpenMP threads for <code>\$eval()</code> . When <code>NULL</code> , uses the maximum available OpenMP thread count.
<code>weights</code>	Optional numeric vector of length <code>nrow(x)</code> .

Value

An object of class "admvn_sun22_tape" with `eval()` and `optim_fns()` (cached `fn/gr`).

dsun22_hs	<i>SUN(2,2) density with joint hyperspherical parameterization</i>
-----------	--

Description

SUN(2,2) density with joint hyperspherical parameterization

Usage

```
dsun22_hs(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

x	Numeric vector of length 2 or an $n \times 2$ matrix of observations.
par	Numeric vector of length 10; see make_sun22_hs_params() .
log	If TRUE (default), return the log-density / log-likelihood.
deriv	Integer derivative order: 0 (default) value only, 1 also gradient, 2 also Hessian, all with respect to par.
n_points, n_shifts, seed	Quasi-Monte Carlo settings for orthant CDFs.
n_threads	Number of OpenMP threads for parallel shard evaluation.
weights	Optional numeric vector of length nrow(x).

dsun22_hs_fun	<i>Create a reusable SUN(2,2) hyperspherical likelihood tape</i>
---------------	--

Description

Create a reusable SUN(2,2) hyperspherical likelihood tape

Usage

```
dsun22_hs_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

x	Numeric vector of length 2 or an $n \times 2$ matrix of observations.
par_seed	Seed vector for make_sun22_hs_params() .
n_points, n_shifts, seed	Quasi-Monte Carlo settings for orthant CDFs.
n_threads	Default OpenMP threads for <code>eval()</code> . When NULL, uses the maximum available OpenMP thread count.
weights	Optional numeric vector of length <code>nrow(x)</code> .

dsun32_hs

SUN(3,2) density with joint hyperspherical parameterization

Description

SUN(3,2) density with joint hyperspherical parameterization

Usage

```
dsun32_hs(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric matrix with 3 columns (observations in rows).
<code>par</code>	Numeric vector of length 16; see make_sun32_hs_params() .
<code>log</code>	Logical; return log-density / log-likelihood if TRUE.
<code>deriv</code>	Integer 0, 1, or 2 for value / gradient / Hessian.
<code>n_points, n_shifts</code>	Orthant QMC settings.
<code>seed</code>	QMC seed.
<code>n_threads</code>	OpenMP threads.
<code>weights</code>	Optional observation weights (length <code>nrow(x)</code>).

<code>dsun32_hs_fun</code>	<i>Create a reusable SUN(3,2) hyperspherical likelihood tape</i>
----------------------------	--

Description

Create a reusable SUN(3,2) hyperspherical likelihood tape

Usage

```
dsun32_hs_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric matrix with 3 columns.
<code>par_seed</code>	Seed vector for make_sun32_hs_params() .
<code>n_points, n_shifts</code>	Orthant QMC settings.
<code>seed</code>	QMC seed.
<code>n_threads</code>	OpenMP threads (default from package helper).
<code>weights</code>	Optional observation weights.

dsun42_hs	<i>SUN(4,2) density with joint hyperspherical parameterization</i>
-----------	--

Description

SUN(4,2) density with joint hyperspherical parameterization

Usage

```
dsun42_hs(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

x	Numeric matrix with 4 columns (observations in rows).
par	Numeric vector of length 23; see make_sun42_hs_params() .
log	Logical; return log-density / log-likelihood if TRUE.
deriv	Integer 0, 1, or 2 for value / gradient / Hessian.
n_points, n_shifts	Orthant QMC settings.
seed	QMC seed.
n_threads	OpenMP threads.
weights	Optional observation weights (length nrow(x)).

dsun42_hs_fun	<i>Create a reusable SUN(4,2) hyperspherical likelihood tape</i>
---------------	--

Description

Create a reusable SUN(4,2) hyperspherical likelihood tape

Usage

```
dsun42_hs_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

x	Numeric matrix with 4 columns.
par_seed	Seed vector for make_sun42_hs_params() .
n_points, n_shifts	Orthant QMC settings.
seed	QMC seed.
n_threads	OpenMP threads (default from package helper).
weights	Optional observation weights.

dsun43_hs

SUN(4,3) density with joint hyperspherical parameterization

Description

SUN(4,3) density with joint hyperspherical parameterization

Usage

```
dsun43_hs(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

x	Numeric matrix with 4 columns (observations in rows).
par	Numeric vector of length 29; see make_sun43_hs_params() .
log	Logical; return log-density / log-likelihood if TRUE.
deriv	Integer 0, 1, or 2 for value / gradient / Hessian.
n_points, n_shifts	Orthant QMC settings.
seed	QMC seed.
n_threads	OpenMP threads.
weights	Optional observation weights (length nrow(x)).

dsun43_hs_fun	<i>Create a reusable SUN(4,3) hyperspherical likelihood tape</i>
---------------	--

Description

Create a reusable SUN(4,3) hyperspherical likelihood tape

Usage

```
dsun43_hs_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

x	Numeric matrix with 4 columns.
par_seed	Seed vector for make_sun43_hs_params() .
n_points, n_shifts	Orthant QMC settings.
seed	QMC seed.
n_threads	OpenMP threads (default from package helper).
weights	Optional observation weights.

dsun44

*SUN(4,4) density with derivatives***Description**

Evaluates the log-density (or density) of a SUN(4,4) distribution using the block-Cholesky parameterization from `make_sun44_params()`. For a matrix of observations the returned value is the summed log-likelihood, or a weighted sum if `weights` is supplied. Derivatives are with respect to the 36-component parameter vector only.

Usage

```
dsun44(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric vector of length 4 or an $n \times 4$ matrix of observations.
<code>par</code>	Numeric vector of length 36; see make_sun44_params() .
<code>log</code>	If TRUE (default), return the log-density / log-likelihood.
<code>deriv</code>	Integer derivative order: 0 (default) value only, 1 also gradient, 2 also Hessian, all with respect to <code>par</code> .
<code>n_points, n_shifts, seed</code>	Quasi-Monte Carlo settings for orthant CDFs.
<code>n_threads</code>	Number of OpenMP threads for parallel shard evaluation.
<code>weights</code>	Optional numeric vector of length <code>nrow(x)</code> .

Details

Orthant probabilities use Genz QMC with CppAD (dimension 4), so prefer a modest `n_points/n_shifts` budget for speed.

Value

A list with value and error. If `deriv >= 1`, also gradient (length 36). If `deriv >= 2`, also hessian (36×36).

See Also

[make_sun44_params\(\)](#), [dsun44_fun\(\)](#), [dsun\(\)](#)

dsun44_fun	<i>Create a reusable SUN(4,4) log-likelihood tape</i>
------------	---

Description

Builds per-observation CppAD shard tapes for the (weighted) SUN(4,4) log-likelihood with data x fixed. Use for KL / MLE: build once, then call `$eval(par)` or `$optim_fns()`.

Usage

```
dsun44_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric vector of length 4 or an $n \times 4$ matrix of observations.
<code>par_seed</code>	Seed parameter vector used when building the tape.
<code>n_points, n_shifts, seed</code>	Quasi-Monte Carlo settings for orthant CDFs.
<code>n_threads</code>	Default OpenMP threads for <code>\$eval()</code> . When NULL, uses the maximum available OpenMP thread count.
<code>weights</code>	Optional numeric vector of length <code>nrow(x)</code> .

Value

An object of class "admvn_sun44_tape" with `eval()` and `optim_fns()` (cached fn/gr).

dsun44_hs

*SUN(4,4) density with joint hyperspherical parameterization***Description**

SUN(4,4) density with joint hyperspherical parameterization

Usage

```
dsun44_hs(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)
```

Arguments

x	Numeric vector of length 4 or an $n \times 4$ matrix of observations.
par	Numeric vector of length 36; see make_sun44_hs_params() .
log	If TRUE (default), return the log-density / log-likelihood.
deriv	Integer derivative order: 0 (default) value only, 1 also gradient, 2 also Hessian, all with respect to par.
n_points, n_shifts, seed	Quasi-Monte Carlo settings for orthant CDFs.
n_threads	Number of OpenMP threads for parallel shard evaluation.
weights	Optional numeric vector of length nrow(x).

dsun44_hs_fun

*Create a reusable SUN(4,4) hyperspherical likelihood tape***Description**

Create a reusable SUN(4,4) hyperspherical likelihood tape

Usage

```

dsun44_hs_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)

```

Arguments

x	Numeric vector of length 4 or an $n \times 4$ matrix of observations.
par_seed	Seed vector for make_sun44_hs_params() .
n_points, n_shifts, seed	Quasi-Monte Carlo settings for orthant CDFs.
n_threads	Default OpenMP threads for <code>eval()</code> . When NULL, uses the maximum available OpenMP thread count.
weights	Optional numeric vector of length <code>nrow(x)</code> .

dsun52_hs

SUN(5,2) density with joint hyperspherical parameterization

Description

SUN(5,2) density with joint hyperspherical parameterization

Usage

```

dsun52_hs(
  x,
  par,
  log = TRUE,
  deriv = 0L,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = 1L,
  weights = NULL
)

```

Arguments

<code>x</code>	Numeric matrix with 5 columns (observations in rows).
<code>par</code>	Numeric vector of length 31; see make_sun52_hs_params() .
<code>log</code>	Logical; return log-density / log-likelihood if TRUE.
<code>deriv</code>	Integer 0, 1, or 2 for value / gradient / Hessian.
<code>n_points, n_shifts</code>	Orthant QMC settings.
<code>seed</code>	QMC seed.
<code>n_threads</code>	OpenMP threads.
<code>weights</code>	Optional observation weights (length <code>nrow(x)</code>).

<code>dsun52_hs_fun</code>	<i>Create a reusable SUN(5,2) hyperspherical likelihood tape</i>
----------------------------	--

Description

Create a reusable SUN(5,2) hyperspherical likelihood tape

Usage

```
dsun52_hs_fun(
  x,
  par_seed,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L,
  n_threads = NULL,
  weights = NULL
)
```

Arguments

<code>x</code>	Numeric matrix with 5 columns.
<code>par_seed</code>	Seed vector for make_sun52_hs_params() .
<code>n_points, n_shifts</code>	Orthant QMC settings.
<code>seed</code>	QMC seed.
<code>n_threads</code>	OpenMP threads (default from package helper).
<code>weights</code>	Optional observation weights.

evaluate_sun22_slice *Slice KL / cross-entropy of a SUN(2,2) vs a weighted target*

Description

Uses `dsun22_fun()` (not `sn`) so the score matches the AD optim objective.

Usage

```
evaluate_sun22_slice(par, quad, n_points = 64L, n_shifts = 2L, n_threads = 1L)
```

Arguments

<code>par</code>	Length-10 SUN(2,2) parameter vector.
<code>quad</code>	List with <code>x</code> , <code>w</code> , and optionally <code>logp</code> (target log-density at each row of <code>x</code>).
<code>n_points, n_shifts</code>	Orthant QMC settings (defaults 64, 2).
<code>n_threads</code>	OpenMP threads for the density tape.

Value

Named vector `c(kl, neg_Elq)`. `kl` is NA if `logp` is missing.

evaluate_sun32_hs_slice
 Slice KL / cross-entropy of a SUN(3,2) hyperspherical fit

Description

Slice KL / cross-entropy of a SUN(3,2) hyperspherical fit

Usage

```
evaluate_sun32_hs_slice(
  par,
  quad,
  n_points = 64L,
  n_shifts = 2L,
  n_threads = 1L
)
```

Arguments

par	Length-21 hyperspherical parameter vector (see make_sun_hs_params()).
quad	List with x, w, and optionally logp (target log-density at each row of x).
n_points, n_shifts	Orthant QMC settings (defaults 64, 2).
n_threads	OpenMP threads for the density tape.

evaluate_sun33_hs_slice

Slice KL / cross-entropy of a SUN(3,3) hyperspherical vs a weighted target

Description

Like [evaluate_sun33_slice\(\)](#), but uses [dsun_hs_fun\(\)](#) and [make_sun_hs_params\(\)](#).

Usage

```
evaluate_sun33_hs_slice(
  par,
  quad,
  n_points = 64L,
  n_shifts = 2L,
  n_threads = 1L
)
```

Arguments

par	Length-21 hyperspherical parameter vector (see make_sun_hs_params()).
quad	List with x, w, and optionally logp (target log-density at each row of x).
n_points, n_shifts	Orthant QMC settings (defaults 64, 2).
n_threads	OpenMP threads for the density tape.

evaluate_sun33_slice *Slice KL / cross-entropy of a SUN(3,3) vs a weighted target*

Description

Uses `dsun_fun()` (not `sn`) so the score matches the AD optim objective.

Usage

```
evaluate_sun33_slice(par, quad, n_points = 64L, n_shifts = 2L, n_threads = 1L)
```

Arguments

<code>par</code>	Length-21 SUN(3,3) parameter vector.
<code>quad</code>	List with <code>x</code> , <code>w</code> , and optionally <code>logp</code> (target log-density at each row of <code>x</code>).
<code>n_points, n_shifts</code>	Orthant QMC settings (defaults 64, 2).
<code>n_threads</code>	OpenMP threads for the density tape.

Value

Named vector `c(kl, neg_Elq)`. `kl` is NA if `logp` is missing.

evaluate_sun42_hs_slice
 Slice KL / cross-entropy of a SUN(4,2) hyperspherical fit

Description

Slice KL / cross-entropy of a SUN(4,2) hyperspherical fit

Usage

```
evaluate_sun42_hs_slice(
  par,
  quad,
  n_points = 64L,
  n_shifts = 2L,
  n_threads = 1L
)
```

Arguments

par	Length-21 hyperspherical parameter vector (see make_sun_hs_params()).
quad	List with x, w, and optionally logp (target log-density at each row of x).
n_points, n_shifts	Orthant QMC settings (defaults 64, 2).
n_threads	OpenMP threads for the density tape.

evaluate_sun43_hs_slice

Slice KL / cross-entropy of a SUN(4,3) hyperspherical fit

Description

Slice KL / cross-entropy of a SUN(4,3) hyperspherical fit

Usage

```
evaluate_sun43_hs_slice(
  par,
  quad,
  n_points = 64L,
  n_shifts = 2L,
  n_threads = 1L
)
```

Arguments

par	Length-21 hyperspherical parameter vector (see make_sun_hs_params()).
quad	List with x, w, and optionally logp (target log-density at each row of x).
n_points, n_shifts	Orthant QMC settings (defaults 64, 2).
n_threads	OpenMP threads for the density tape.

evaluate_sun44_slice *Slice KL / cross-entropy of a SUN(4,4) vs a weighted target*

Description

Uses [dsun44_fun\(\)](#) (not **sn**).

Usage

```
evaluate_sun44_slice(par, quad, n_points = 256L, n_shifts = 4L, n_threads = 1L)
```

Arguments

par	Length-36 SUN(4,4) parameter vector.
quad	List with x, w, and optionally logp (target log-density at each row of x).
n_points, n_shifts	Defaults 256, 4.
n_threads	OpenMP threads for the density tape.

evaluate_sun52_hs_slice

Slice KL / cross-entropy of a SUN(5,2) hyperspherical fit

Description

Slice KL / cross-entropy of a SUN(5,2) hyperspherical fit

Usage

```
evaluate_sun52_hs_slice(
  par,
  quad,
  n_points = 64L,
  n_shifts = 2L,
  n_threads = 1L
)
```

Arguments

par	Length-21 hyperspherical parameter vector (see make_sun_hs_params()).
quad	List with x, w, and optionally logp (target log-density at each row of x).
n_points, n_shifts	Orthant QMC settings (defaults 64, 2).
n_threads	OpenMP threads for the density tape.

fit_sun22_quad

Fit SUN(2,2) by minimising KL on a weighted quadrature

Description

Minimises the weighted cross-entropy $-\sum_i w_i \log q(x_i)$ with analytic gradients from [dsun22_fun\(\)](#).

Usage

```
fit_sun22_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
  mc.cores = 1L,
  verbose = FALSE,
  mvquad_opts = list(),
  optim_opts = list()
)
```

Arguments

quad	List with x ($n \times 2$), w , and optionally $\log p$ (for returned KL).
start	Length-10 start (see make_sun22_start_from_normal()).
lower, upper	Box constraints (defaults from sun22_bounds()).
obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set <code>optim trace = 1</code> .
mvquad_opts	Named list; used keys <code>n_points</code> , <code>n_shifts</code> for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into <code>stats::optim</code> L-BFGS-B control.

Value

A list with `fit`, `par`, `dp`, `start`, `gradient` (objective-scale at `par`), KL / cross-entropy summaries, and `engine`.

fit_sun32_hs_quad	<i>Fit SUN(3,2) hyperspherical by minimising KL on a weighted quadrature</i>
-------------------	--

Description

Fit SUN(3,2) hyperspherical by minimising KL on a weighted quadrature

Usage

```
fit_sun32_hs_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
```

```

    mc.cores = 1L,
    verbose = FALSE,
    mvquad_opts = list(),
    optim_opts = list()
)

```

Arguments

quad	List with x ($n \times 3$), w , and optionally $\log p$ (for returned KL).
start	Length-16 hyperspherical start (see make_sun32_hs_start_from_normal()).
lower, upper	Box constraints (defaults from sun32_hs_bounds()).
obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set optim trace = 1.
mvquad_opts	Named list; used keys <code>n_points</code> , <code>n_shifts</code> for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into <code>stats::optim</code> L-BFGS-B control.

fit_sun33_hs_quad	<i>Fit SUN(3,3) hyperspherical by minimising KL on a weighted quadrature</i>
-------------------	--

Description

Like [fit_sun33_quad\(\)](#), but uses [dsun_hs_fun\(\)](#) and defaults from [sun33_hs_bounds\(\)](#) / [make_sun33_hs_start_from_r](#)

Usage

```

fit_sun33_hs_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
  mc.cores = 1L,
  verbose = FALSE,
  mvquad_opts = list(),
  optim_opts = list()
)

```

Arguments

quad	List with x ($n \times 3$), w , and optionally $\log p$ (for returned KL).
start	Length-21 hyperspherical start (see make_sun33_hs_start_from_normal()).
lower, upper	Box constraints (defaults from sun33_hs_bounds()).

obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set optim trace = 1.
mvquad_opts	Named list; used keys n_points, n_shifts for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into stats::optim L-BFGS-B control.

fit_sun33_quad	<i>Fit SUN(3,3) by minimising KL on a weighted quadrature</i>
----------------	---

Description

Minimises the weighted cross-entropy $-\sum_i w_i \log q(x_i)$ with analytic gradients from [dsun_fun\(\)](#).

Usage

```
fit_sun33_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
  mc.cores = 1L,
  verbose = FALSE,
  mvquad_opts = list(),
  optim_opts = list()
)
```

Arguments

quad	List with x ($n \times 3$), w , and optionally $\log p$ (for returned KL).
start	Length-21 start (see make_sun33_start_from_normal()).
lower, upper	Box constraints (defaults from sun33_bounds()).
obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set optim trace = 1.
mvquad_opts	Named list; used keys n_points, n_shifts for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into stats::optim L-BFGS-B control.

Value

A list with fit, par, dp, start, gradient (objective-scale at par), KL / cross-entropy summaries, and engine.

fit_sun42_hs_quad	<i>Fit SUN(4,2) hyperspherical by minimising KL on a weighted quadrature</i>
-------------------	--

Description

Fit SUN(4,2) hyperspherical by minimising KL on a weighted quadrature

Usage

```
fit_sun42_hs_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
  mc.cores = 1L,
  verbose = FALSE,
  mvquad_opts = list(),
  optim_opts = list()
)
```

Arguments

quad	List with x ($n \times 3$), w , and optionally $\log p$ (for returned KL).
start	Length-23 hyperspherical start (see make_sun42_hs_start_from_normal()).
lower, upper	Box constraints (defaults from sun42_hs_bounds()).
obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set <code>optim trace = 1</code> .
mvquad_opts	Named list; used keys <code>n_points</code> , <code>n_shifts</code> for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into <code>stats::optim</code> L-BFGS-B control.

fit_sun43_hs_quad	<i>Fit SUN(4,3) hyperspherical by minimising KL on a weighted quadrature</i>
-------------------	--

Description

Fit SUN(4,3) hyperspherical by minimising KL on a weighted quadrature

Usage

```
fit_sun43_hs_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
  mc.cores = 1L,
  verbose = FALSE,
  mvquad_opts = list(),
  optim_opts = list()
)
```

Arguments

quad	List with x ($n \times 3$), w , and optionally $\log p$ (for returned KL).
start	Length-29 hyperspherical start (see make_sun43_hs_start_from_normal()).
lower, upper	Box constraints (defaults from sun43_hs_bounds()).
obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set optim trace = 1.
mvquad_opts	Named list; used keys <code>n_points</code> , <code>n_shifts</code> for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into <code>stats::optim</code> L-BFGS-B control.

fit_sun44_quad	<i>Fit SUN(4,4) by minimising KL on a weighted quadrature</i>
----------------	---

Description

Like [fit_sun33_quad\(\)](#) for SUN(4,4) via [dsun44_fun\(\)](#). Default QMC budget is `n_points = 256`, `n_shifts = 4`.

Usage

```
fit_sun44_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
  mc.cores = 1L,
  verbose = FALSE,
  mvquad_opts = list(),
  optim_opts = list()
)
```

Arguments

quad	List with x ($n \times 3$), w , and optionally $\log p$ (for returned KL).
start	Length-36 start (see make_sun44_start_from_normal()).
lower, upper	Box constraints (defaults from sun33_bounds()).
obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set <code>optim trace = 1</code> .
mvquad_opts	Named list; used keys <code>n_points</code> , <code>n_shifts</code> for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into <code>stats::optim</code> L-BFGS-B control.

fit_sun52_hs_quad	<i>Fit SUN(5,2) hyperspherical by minimising KL on a weighted quadrature</i>
-------------------	--

Description

Fit SUN(5,2) hyperspherical by minimising KL on a weighted quadrature

Usage

```
fit_sun52_hs_quad(
  quad,
  start,
  lower = NULL,
  upper = NULL,
  obj_scale = 10000,
  mc.cores = 1L,
  verbose = FALSE,
  mvquad_opts = list(),
  optim_opts = list()
)
```

Arguments

quad	List with x ($n \times 5$), w , and optionally $\log p$ (for returned KL).
start	Length-31 hyperspherical start (see make_sun52_hs_start_from_normal()).
lower, upper	Box constraints (defaults from sun52_hs_bounds()).
obj_scale	Scale applied to the optim objective (default 1e4).
mc.cores	OpenMP threads for the density tape.
verbose	If TRUE, set <code>optim trace = 1</code> .
mvquad_opts	Named list; used keys <code>n_points</code> , <code>n_shifts</code> for orthant QMC (defaults 64, 2).
optim_opts	Named list merged into <code>stats::optim</code> L-BFGS-B control.

gh_points

*Gauss-Hermite nodes and weights under a multivariate normal***Description**

Product Gauss-Hermite quadrature for $N(\mu, \Sigma)$, returned as abscissae in the original scale and corresponding product weights.

Usage

```
gh_points(mu, Sigma, n = 15)
```

Arguments

mu	Mean vector.
Sigma	Covariance matrix (made symmetric; Cholesky used for the map).
n	Number of univariate GH nodes per dimension (default 15).

Value

A list with x ($n^d \times d$ matrix) and w (length n^d weights).

make_sun_hs_params

*Build SUN(3,3) dp from a joint hyperspherical parameter vector***Description**

Alternative to [make_sun_params\(\)](#): the latent joint correlation of $(U_*, V) \in \mathbb{R}^6$ is $J = BB^\top$ with unit-row Cholesky factor B from fifteen unconstrained coordinates z_{ij} via $t_{ij} = \tanh(z_{ij})$. Then $C_u = J_{UU}$, $\Delta = J_{UV}$, $\Gamma = J_{VV}$, and $\Omega = D_u C_u D_u$ with $D_u = \text{diag}(\nu)$.

Usage

```
make_sun_hs_params(par)
```

Arguments

par	Numeric vector of length 21: $\text{xi}(3)$, $\text{nu}(3)$, then free z in row-major fill order $z_{21}, z_{31}, z_{32}, z_{41}, \dots, z_{65}$. nu are marginal SDs of Ω ($\Omega_{jj} = \nu_j^2$).
-----	--

Details

The leading six parameters match the block-Cholesky layout (xi, nu), so the C++ sun_mle() driver works unchanged on hyperspherical tapes.

Independent skew-normals. Only the pair slots $z_{4,1}, z_{5,2}, z_{6,3}$ nonzero (equal to $\operatorname{atanh}(d_i)$) yields $J = \begin{bmatrix} I & D \\ D & I \end{bmatrix}$ with $D = \operatorname{diag}(d)$. All $z = 0$ recovers a Gaussian.

Fill order. For dimension $d = 6$, free entries are filled by rows $i = 2, \dots, 6$, columns $j = 1, \dots, i - 1$.

Value

A list with components xi, Omega, Delta, tau (fixed at zero), and Gamma.

See Also

[make_sun_params\(\)](#), [dsun_hs\(\)](#), [sun33_hs_bounds\(\)](#)

Examples

```
par <- c(
  xi1 = 0, xi2 = 0, xi3 = 0,
  nu1 = 1, nu2 = 1, nu3 = 1,
  z21 = 0, z31 = 0, z32 = 0,
  z41 = atanh(0.4), z42 = 0, z43 = 0,
  z51 = 0, z52 = atanh(0.3), z53 = 0, z54 = 0,
  z61 = 0, z62 = 0, z63 = atanh(0.2), z64 = 0, z65 = 0
)
dp <- make_sun_hs_params(par)
diag(dp$Gamma)
```

make_sun_params

Build SUN distribution parameters from a free parameter vector

Description

Converts the SUN(3,3) block-Cholesky parameterization used in the supervision notes into the dp list expected by the **sn** package (xi, Omega, Delta, tau, Gamma).

Usage

```
make_sun_params(par)
```

Arguments

par Numeric vector of length 21: $c(\text{xi1}, \text{xi2}, \text{xi3}, \text{nu1}, \text{nu2}, \text{nu3}, \text{omega12}, \text{omega13}, \text{omega23}, \text{L11}, \text{L12}, \text{L13}, \text{L21}, \text{L22}, \text{L23}, \text{L31}, \text{L32}, \text{L33}, \text{gamma12}, \text{gamma13}, \text{gamma23})$. Here nu are marginal standard deviations of Ω ($\Omega_{jj} = \nu_j^2$), and omega^* are unconstrained hyperspherical coordinates for the correlation C_u of the standardized latent. Skew loadings L act on that correlation scale (all nine entries free; diagonals typically in $[-1, 1]$, off-diagonals in $[-0.5, 0.5]$). gamma^* parameterize the residual correlation of V . Valid parameters need $\text{diag}(LC_u L^\top) < 1$.

Details

Correlation-scale SUN, then scale. With correlation C_u from the unit-row Cholesky of $(\omega_{12}, \omega_{13}, \omega_{23})$ and marginal standard deviations $\nu_j > 0$, writing $D_u = \text{diag}(\nu)$,

$$\Omega = D_u C_u D_u.$$

Skew loadings L multiply C_u (not Ω):

$$M = LC_u L^\top, \quad \Sigma_{uv} = D_u C_u L^\top.$$

Free $(\gamma_{12}, \gamma_{13}, \gamma_{23})$ build residual correlation C . With $D = \text{diag}(1 - \text{diag}(M))$,

$$R = D^{1/2} C D^{1/2},$$

so $\Sigma_{vv} = M + R$ is a correlation matrix. Then $\Gamma = \Sigma_{vv}$ and $\Delta = D_u^{-1} \Sigma_{uv} = C_u L^\top$.

Skew loadings. All nine entries of L are free:

$$L = \begin{bmatrix} L_{11} & L_{12} & L_{13} \\ L_{21} & L_{22} & L_{23} \\ L_{31} & L_{32} & L_{33} \end{bmatrix}.$$

Value

A list with components xi , Ω , Δ , τ (fixed at zero), and Γ .

Examples

```
par <- c(
  xi1 = 0, xi2 = 0, xi3 = 0,
  nu1 = 1, nu2 = 1, nu3 = 1,
  omega12 = 0.2, omega13 = 0.1, omega23 = 0.2,
  L11 = 0.4, L12 = 0.1, L13 = 0.1,
  L21 = 0.1, L22 = 0.4, L23 = 0.1,
  L31 = 0, L32 = 0.1, L33 = 0.35,
  gamma12 = 0.3, gamma13 = 0.2, gamma23 = 0.3
)
make_sun_params(par)
```

make_sun22_hs_params *Build SUN(2,2) parameters from a joint hyperspherical vector*

Description

The joint correlation of (U, V) is formed from six unconstrained unit-row Cholesky coordinates in row-major order.

Usage

```
make_sun22_hs_params(par)
```

Arguments

par Numeric vector of length 10: xi(2), nu(2), then z21, z31, z32, z41, z42, z43.

Value

A SUN dp list with xi, Omega, Delta, zero tau, and Gamma.

See Also

[dsun22_hs\(\)](#), [make_sun22_params\(\)](#), [sun22_hs_bounds\(\)](#)

make_sun22_hs_start_from_gh *SUN(2,2) hyperspherical start from weighted sample moments*

Description

SUN(2,2) hyperspherical start from weighted sample moments

Usage

```
make_sun22_hs_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun22_start_from_normal\(\)](#).

Value

Named length-10 parameter vector.

make_sun22_hs_start_from_normal

Map mean and covariance to a SUN(2,2) hyperspherical start

Description

Map mean and covariance to a SUN(2,2) hyperspherical start

Usage

```
make_sun22_hs_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

mu	Length-2 mean.
Sigma	2×2 covariance.
skew_strength	Diagonal loading for L^* (default 0).

Value

Named length-10 vector for [make_sun22_hs_params\(\)](#).

make_sun22_params

Build SUN(2,2) distribution parameters from a free parameter vector

Description

Converts the SUN(2,2) block-Cholesky parameterization into the dp list expected by the **sn** package (xi, Omega, Delta, tau, Gamma).

Usage

```
make_sun22_params(par)
```

Arguments

par	Numeric vector of length 10: xi(2), nu(2), omega12(1), L11, L12, L21, L22, gamma12. nu are marginal standard deviations of Ω ($\Omega_{jj} = \nu_j^2$); omega12 free hyperspherical coordinate for correlation C_u ; all four entries of L are free on the correlation scale; gamma12 parameterizes the residual correlation of V . Valid parameters need $\text{diag}(LC_u L^\top) < 1$.
-----	---

Details

As in [make_sun_params\(\)](#): $\Omega = D_u C_u D_u$ with $D_u = \text{diag}(\nu)$, $M = LC_u L^\top$, $\Sigma_{uv} = D_u C_u L^\top$, and $\Sigma_{vv} = M + D^{1/2} C D^{1/2}$ with unit diagonal.

Value

A list with components xi, Omega, Delta, tau (fixed at zero), and Gamma.

See Also

[make_sun_params\(\)](#), [dsun22\(\)](#), [dsun22_fun\(\)](#)

make_sun22_start_from_gh

SUN(2,2) start from weighted sample moments on a quadrature

Description

SUN(2,2) start from weighted sample moments on a quadrature

Usage

```
make_sun22_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun22_start_from_normal\(\)](#).

Value

Named length-10 parameter vector.

make_sun22_start_from_normal

Map mean and covariance to a SUN(2,2) start (zero / tiny skewness)

Description

Uses marginal standard deviations and unconstrained correlation free parameters for (nu, omega). Skew loadings and residual-correlation free parameters are zero, or a tiny diagonal loading if skew_strength > 0.

Usage

```
make_sun22_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

mu Length-2 mean.
 Sigma 2×2 covariance.
 skew_strength Diagonal loading for L^* (default 0).

Value

Named length-10 parameter vector for [make_sun22_params\(\)](#).

make_sun32_hs_params *Build SUN(3,2) parameters from a joint hyperspherical vector*

Description

The joint correlation of $(U, V) \in \mathbb{R}^5$ is formed from ten unconstrained unit-row Cholesky coordinates in row-major order.

Usage

```
make_sun32_hs_params(par)
```

Arguments

par Numeric vector of length 16: xi(3), nu(3), then z21, z31, z32, z41, z42, z43, z51, z52, z53, z54.

Value

A SUN dp list with xi, Omega (3×3), Delta (3×2), zero tau (length 2), and Gamma (2×2).

See Also

[dsun32_hs\(\)](#), [sun32_hs_bounds\(\)](#)

make_sun32_hs_start_from_gh *SUN(3,2) hyperspherical start from weighted sample moments*

Description

SUN(3,2) hyperspherical start from weighted sample moments

Usage

```
make_sun32_hs_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun32_hs_start_from_normal\(\)](#).

Value

Named length-16 parameter vector.

make_sun32_hs_start_from_normal

Map mean and covariance to a SUN(3,2) hyperspherical start

Description

Exact Gaussian start on the joint hyperspherical layout of [make_sun32_hs_params\(\)](#): the three U -block free coordinates come from inverting `cov2cor(Sigma)`; all cross / V coordinates are zero. Optional skew_strength sets pair slots z41, z52 to `atanh(skew_strength)`.

Usage

`make_sun32_hs_start_from_normal(mu, Sigma, skew_strength = 0)`

Arguments

mu Length-3 mean.
 Sigma 3×3 covariance.
 skew_strength Pair-slot $t = \tanh(z)$ value in $(-1, 1)$ (default 0).

Value

Named length-16 parameter vector for [make_sun32_hs_params\(\)](#).

make_sun33_hs_start_from_gh

SUN(3,3) hyperspherical start from weighted sample moments

Description

SUN(3,3) hyperspherical start from weighted sample moments

Usage

`make_sun33_hs_start_from_gh(quad, skew_strength = 0.01)`

Arguments

quad List with x and w .
 skew_strength Passed to `make_sun33_hs_start_from_normal()`.

Value

Named length-21 parameter vector.

`make_sun33_hs_start_from_normal`

Map mean and covariance to a $SUN(3,3)$ hyperspherical start

Description

Exact Gaussian start on the joint hyperspherical layout of `make_sun_hs_params()`: the three U -block free coordinates come from inverting `cov2cor(Sigma)`; all cross / V coordinates are zero (so $J = \text{blockdiag}(C_u, I)$). Optional `skew_strength` sets the independent-SN pair slots to $\text{atanh}(\text{skew_strength})$.

Usage

```
make_sun33_hs_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

`mu` Length-3 mean.
`Sigma` 3×3 covariance.
`skew_strength` Pair-slot $t = \tanh(z)$ value in $(-1, 1)$ (default 0).

Value

Named length-21 parameter vector for `make_sun_hs_params()`.

`make_sun33_start_from_gh`

$SUN(3,3)$ start from weighted sample moments on a quadrature

Description

$SUN(3,3)$ start from weighted sample moments on a quadrature

Usage

```
make_sun33_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun33_start_from_normal\(\)](#).

Value

Named length-21 parameter vector.

make_sun33_start_from_normal

Map mean and covariance to a SUN(3,3) start (zero / tiny skewness)

Description

Uses marginal standard deviations and unconstrained correlation free parameters for (nu, omega). Skew loadings and residual-correlation free parameters are zero, or a tiny diagonal loading if skew_strength > 0.

Usage

```
make_sun33_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

mu Length-3 mean.
 Sigma 3×3 covariance.
 skew_strength Diagonal loading for L* (default 0).

Value

Named length-21 parameter vector for [make_sun_params\(\)](#).

make_sun42_hs_params *Build SUN(4,2) parameters from a joint hyperspherical vector*

Description

The joint correlation of $(U, V) \in \mathbb{R}^6$ is formed from 15 unconstrained unit-row Cholesky coordinates in row-major order.

Usage

```
make_sun42_hs_params(par)
```

Arguments

par Numeric vector of length 23: xi(4), nu(4), then 15 z coordinates named by .sun_hs_z_names(4, 2).

Value

A SUN dp list with xi, Omega (4×4), Delta (4×2), zero tau (length 2), and Gamma (2×2).

See Also

[dsun42_hs\(\)](#), [sun42_hs_bounds\(\)](#)

make_sun42_hs_start_from_gh

SUN(4,2) hyperspherical start from weighted sample moments

Description

SUN(4,2) hyperspherical start from weighted sample moments

Usage

```
make_sun42_hs_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun42_hs_start_from_normal\(\)](#).

Value

Named length-23 parameter vector.

make_sun42_hs_start_from_normal

Map mean and covariance to a SUN(4,2) hyperspherical start

Description

Exact Gaussian start on the joint hyperspherical layout of [make_sun42_hs_params\(\)](#): the six U -block free coordinates come from inverting cov2cor(Sigma); all cross / V coordinates are zero. Optional skew_strength sets pair slots z51, z62 to atanh(skew_strength).

Usage

```
make_sun42_hs_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

mu	Length-4 mean.
Sigma	4×4 covariance.
skew_strength	Pair-slot $t = \tanh(z)$ value in $(-1, 1)$ (default 0).

Value

Named length-23 parameter vector for [make_sun42_hs_params\(\)](#).

make_sun43_hs_params *Build SUN(4,3) parameters from a joint hyperspherical vector*

Description

The joint correlation of $(U, V) \in \mathbb{R}^7$ is formed from 21 unconstrained unit-row Cholesky coordinates in row-major order.

Usage

```
make_sun43_hs_params(par)
```

Arguments

par	Numeric vector of length 29: xi(4), nu(4), then 21 z coordinates named by .sun_hs_z_names(4, 3).
-----	--

Value

A SUN dp list with xi, Omega (4×4), Delta (4×3), zero tau (length 3), and Gamma (3×3).

See Also

[dsun43_hs\(\)](#), [sun43_hs_bounds\(\)](#)

make_sun43_hs_start_from_gh

SUN(4,3) hyperspherical start from weighted sample moments

Description

SUN(4,3) hyperspherical start from weighted sample moments

Usage

```
make_sun43_hs_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun43_hs_start_from_normal\(\)](#).

Value

Named length-29 parameter vector.

make_sun43_hs_start_from_normal

Map mean and covariance to a SUN(4,3) hyperspherical start

Description

Exact Gaussian start on the joint hyperspherical layout of [make_sun43_hs_params\(\)](#): the six U -block free coordinates come from inverting `cov2cor(Sigma)`; all cross / V coordinates are zero. Optional `skew_strength` sets pair slots `z51`, `z62`, `z73` to `atanh(skew_strength)`.

Usage

```
make_sun43_hs_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

mu Length-4 mean.
 Sigma 4×4 covariance.
 skew_strength Pair-slot $t = \tanh(z)$ value in $(-1, 1)$ (default 0).

Value

Named length-29 parameter vector for [make_sun43_hs_params\(\)](#).

make_sun44_hs_params *Build SUN(4,4) parameters from a joint hyperspherical vector*

Description

The joint correlation of (U, V) is formed from 28 unconstrained unit-row Cholesky coordinates in row-major order.

Usage

```
make_sun44_hs_params(par)
```

Arguments

par Numeric vector of length 36: xi(4), nu(4), then 28 z coordinates named by .sun_hs_z_names(4, 4).

Value

A SUN dp list with xi, Omega, Delta, zero tau, and Gamma.

See Also

[dsun44_hs\(\)](#), [make_sun44_params\(\)](#), [sun44_hs_bounds\(\)](#)

make_sun44_hs_start_from_gh *SUN(4,4) hyperspherical start from weighted sample moments*

Description

SUN(4,4) hyperspherical start from weighted sample moments

Usage

```
make_sun44_hs_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun33_start_from_normal\(\)](#).

Value

Named length-36 parameter vector.

make_sun44_hs_start_from_normal

Map mean and covariance to a SUN(4,4) hyperspherical start

Description

Map mean and covariance to a SUN(4,4) hyperspherical start

Usage

```
make_sun44_hs_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

mu	Length-4 mean.
Sigma	4×4 covariance.
skew_strength	Diagonal loading for L* (default 0).

Value

Named length-36 vector for [make_sun44_hs_params\(\)](#).

make_sun44_params

Build SUN(4,4) distribution parameters from a free parameter vector

Description

Converts the SUN(4,4) block-Cholesky parameterization into the dp list expected by the **sn** package (xi, Omega, Delta, tau, Gamma).

Usage

```
make_sun44_params(par)
```

Arguments

par	Numeric vector of length 36: xi(4), nu(4), omega(6), L(16), gamma(6). nu are marginal standard deviations of Ω ($\Omega_{jj} = \nu_j^2$); omega* free hyperspherical coordinates for correlation C_u ; L acts on that correlation scale; gamma* parameterize the residual correlation of V. Valid parameters need $\text{diag}(LC_u L^\top) < 1$.
-----	---

Details

As in [make_sun_params\(\)](#): $\Omega = D_u C_u D_u$ with $D_u = \text{diag}(\nu)$, $M = LC_u L^\top$, $\Sigma_{uv} = D_u C_u L^\top$, and $\Sigma_{vv} = M + D^{1/2} C D^{1/2}$ with unit diagonal.

Value

A list with components xi, Omega, Delta, tau (fixed at zero), and Gamma.

See Also

[make_sun_params\(\)](#), [dsun44\(\)](#), [dsun44_fun\(\)](#)

make_sun44_start_from_gh

SUN(4,4) start from weighted sample moments on a quadrature

Description

SUN(4,4) start from weighted sample moments on a quadrature

Usage

```
make_sun44_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
 skew_strength Passed to [make_sun33_start_from_normal\(\)](#).

Value

Named length-36 parameter vector.

make_sun44_start_from_normal

Map mean and covariance to a SUN(4,4) start (zero / tiny skewness)

Description

Map mean and covariance to a SUN(4,4) start (zero / tiny skewness)

Usage

```
make_sun44_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

mu Length-4 mean.
 Sigma 4×4 covariance.
 skew_strength Diagonal loading for L* (default 0).

Value

Named length-36 parameter vector for [make_sun44_params\(\)](#).

make_sun52_hs_params *Build SUN(5,2) parameters from a joint hyperspherical vector*

Description

The joint correlation of $(U, V) \in \mathbb{R}^7$ is formed from 21 unconstrained unit-row Cholesky coordinates in row-major order.

Usage

```
make_sun52_hs_params(par)
```

Arguments

par Numeric vector of length 31: xi(5), nu(5), then 21 z coordinates named by .sun_hs_z_names(5, 2).

Value

A SUN dp list with xi, Omega (5×5), Delta (5×2), zero tau (length 2), and Gamma (2×2).

See Also

[dsun52_hs\(\)](#), [sun52_hs_bounds\(\)](#)

make_sun52_hs_start_from_gh *SUN(5,2) hyperspherical start from weighted sample moments*

Description

SUN(5,2) hyperspherical start from weighted sample moments

Usage

```
make_sun52_hs_start_from_gh(quad, skew_strength = 0.01)
```

Arguments

quad List with x and w.
skew_strength Passed to [make_sun52_hs_start_from_normal\(\)](#).

Value

Named length-31 parameter vector.

make_sun52_hs_start_from_normal

Map mean and covariance to a SUN(5,2) hyperspherical start

Description

Exact Gaussian start on the joint hyperspherical layout of [make_sun52_hs_params\(\)](#): the ten U -block free coordinates come from inverting `cov2cor(Sigma)`; all cross / V coordinates are zero. Optional skew_strength sets pair slots `z61, z72` to `atanh(skew_strength)`.

Usage

```
make_sun52_hs_start_from_normal(mu, Sigma, skew_strength = 0)
```

Arguments

<code>mu</code>	Length-5 mean.
<code>Sigma</code>	5×5 covariance.
<code>skew_strength</code>	Pair-slot $t = \tanh(z)$ value in $(-1, 1)$ (default 0).

Value

Named length-31 parameter vector for [make_sun52_hs_params\(\)](#).

pack_genz_ch

Pack covariance into Genz chol and scales

Description

Converts a covariance matrix into the Genz reordered correlation Cholesky factor and marginal scales used by the CppAD tape. The permutation `perm` must match the frozen order from [pmvn_fun\(\)](#).

Usage

```
pack_genz_ch(sigma, perm)
```

Arguments

<code>sigma</code>	Positive-definite covariance matrix.
<code>perm</code>	Integer permutation vector from a tape object.

Value

A list with components `scale` and `ch`.

pmvn

*Multivariate normal CDF with derivatives***Description**

Evaluates $P(\text{lower} < X < \text{upper})$ for $X \sim N(\text{mean}, \text{sigma})$ using Genz's quasi-Monte Carlo algorithm. Returns the probability, a Monte Carlo error estimate, and derivatives of the approximation with respect to upper.

Usage

```
pmvn(
  upper,
  lower = -Inf,
  mean = 0,
  sigma,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L
)
```

Arguments

upper	Numeric vector of upper integration limits.
lower	Numeric vector of lower integration limits. Defaults to -Inf.
mean	Numeric mean vector. Defaults to zero.
sigma	Positive-definite covariance matrix.
n_points	Number of quasi-Monte Carlo points per random shift.
n_shifts	Number of random shifts for the QMC rule.
seed	Random seed for generating QMC shifts.

Value

A list with components value, error, gradient, and hessian. The gradient and Hessian are with respect to upper.

Examples

```
sigma <- matrix(c(1, 0.5, 0.5, 1), 2, 2)
pmvn(c(1, 1), lower = c(-Inf, -Inf), sigma = sigma)
```

pmvn_fun

*Create a reusable MVN CDF tape***Description**

Builds a CppAD tape for the multivariate normal CDF over upper, mean, marginal scales, and the Genz Cholesky factor. The tape is built once; subsequent evaluations pass new parameter values without re-taping. Derivatives with respect to upper use sparse inner patterns (as in `adlaplace`).

Usage

```
pmvn_fun(
  lower = -Inf,
  mean = 0,
  sigma,
  n_points = 1021L,
  n_shifts = 8L,
  seed = 1L
)
```

Arguments

lower, mean, sigma

Seed integration parameters used to fix the Genz permutation and build the tape (see `pmvn()`).

n_points, n_shifts, seed

QMC settings passed to `pmvn()`.

Details

`eval(upper, mean = NULL, sigma = NULL, inner = TRUE)` evaluates the taped function. When mean or sigma are omitted, build-time values are used. `inner = TRUE` (default) returns the gradient and Hessian with respect to upper only; `inner = FALSE` also returns `gradient_mean` and includes mean, scale, and Cholesky blocks in the sparse derivatives.

Value

An object of class "admvn_tape" with components `perm` (frozen Genz permutation) and `eval()`.

Examples

```
sigma <- matrix(c(1, 0.3, 0.3, 1), 2, 2)
f <- pmvn_fun(lower = c(-Inf, -Inf), mean = c(0, 0), sigma = sigma)
f$eval(c(0.5, 0.5))
```

<code>print_sun_kl_fit</code>	<i>Print SUN KL fit diagnostics (including gradient)</i>
-------------------------------	--

Description

Print SUN KL fit diagnostics (including gradient)

Usage

```
print_sun_kl_fit(fit, label = NULL)
```

Arguments

<code>fit</code>	Result from <code>fit_sun22_quad()</code> / <code>fit_sun33_quad()</code> / <code>fit_sun33_hs_quad()</code> / <code>fit_sun44_quad()</code> (or a toenail wrapper that preserves the same fields).
<code>label</code>	Optional header string.

Value

`fit`, invisibly.

<code>sun_mle</code>	<i>SUN(3,3) MLE via C++ trust-region optimization</i>
----------------------	---

Description

Maximizes the taped SUN log-likelihood using the quasi-Newton trust-region solver from **trustOptim** (SR1 by default). The nu components are optimized on the log scale so that they remain positive. The entire optimization runs in C++ with one OpenMP-parallel value+gradient evaluation per iteration.

Usage

```
sun_mle(tape, start, control = list(), n_threads = 0L)
```

Arguments

<code>tape</code>	An object from <code>dsun_fun()</code> .
<code>start</code>	Numeric start vector of length 21 (natural parameterization).
<code>control</code>	List of trust-region controls (see trustOptim), e.g. <code>maxit</code> , <code>grad.tol</code> , <code>report.freq</code> , <code>quasi.newton.method</code> (1 = SR1, 2 = BFGS).
<code>n_threads</code>	Threads for shard evaluation; 0 uses the tape default.

Value

A list with `par`, `value` (log-likelihood), `gradient`, `iterations`, `status`, and related fields.

sun22_bounds	<i>Default L-BFGS-B box constraints for SUN(2,2) parameters</i>
--------------	---

Description

Diagonal loadings L11,L22 are in $[-1, 1]$ (correlation scale). Off-diagonal loadings L12,L21 are in $[-0.5, 0.5]$, matching [sun33_bounds\(\)](#).

Usage

```
sun22_bounds()
```

Value

List with lower and upper length-10 numeric vectors.

See Also

[make_sun22_params\(\)](#), [fit_sun22_quad\(\)](#)

sun22_hs_bounds	<i>Default box constraints for SUN(2,2) hyperspherical parameters</i>
-----------------	---

Description

Independent-SN pair slots use $\text{atanh}(\pm 0.8)$ and all other joint coordinates use $\text{atanh}(\pm 0.5)$.

Usage

```
sun22_hs_bounds()
```

Value

List of length-10 lower and upper vectors.

sun32_hs_bounds	<i>Default box constraints for SUN(3,2) hyperspherical parameters</i>
-----------------	---

Description

Pair slots linking V_i to U_i for $i = 1, 2$ (z41, z52) use $\text{atanh}(\pm 0.8)$; other joint coordinates use $\text{atanh}(\pm 0.5)$.

Usage

sun32_hs_bounds()

Value

List of length-16 lower and upper vectors.

sun33_bounds	<i>Default L-BFGS-B box constraints for SUN(3,3) parameters</i>
--------------	---

Description

Diagonal loadings L11, L22, L33 are in $[-1, 1]$ (correlation scale). Off-diagonal loadings L_{ij}/L_{ji} are in $[-0.5, 0.5]$.

Usage

sun33_bounds()

Value

List with lower and upper length-21 numeric vectors.

See Also

[make_sun_params\(\)](#), [fit_sun33_quad\(\)](#)

sun33_hs_bounds	<i>Default L-BFGS-B box constraints for SUN(3,3) hyperspherical parameters</i>
-----------------	--

Description

Soft boxes on the joint unit-row Cholesky coordinates of `make_sun_hs_params()`: independent-SN pair slots `z41`, `z52`, `z63` lie in $\text{atanh}(\pm 0.8)$; all other `z*` lie in $\text{atanh}(\pm 0.5)$. These are regularization / start boxes, not a Gershgorin PD certificate.

Usage

```
sun33_hs_bounds()
```

Value

List with lower and upper length-21 numeric vectors.

See Also

`make_sun_hs_params()`, `dsun_hs()`

sun42_hs_bounds	<i>Default box constraints for SUN(4,2) hyperspherical parameters</i>
-----------------	---

Description

Pair slots linking V_i to U_i for $i = 1, 2$ (`z51`, `z62`) use $\text{atanh}(\pm 0.8)$; other joint coordinates use $\text{atanh}(\pm 0.5)$.

Usage

```
sun42_hs_bounds()
```

Value

List of length-23 lower and upper vectors.

sun43_hs_bounds	<i>Default box constraints for SUN(4,3) hyperspherical parameters</i>
-----------------	---

Description

Pair slots linking V_i to U_i for $i = 1, 2, 3$ (z51, z62, z73) use $\text{atanh}(\pm 0.8)$; other joint coordinates use $\text{atanh}(\pm 0.5)$.

Usage

sun43_hs_bounds()

Value

List of length-29 lower and upper vectors.

sun44_bounds	<i>Default L-BFGS-B box constraints for SUN(4,4) parameters</i>
--------------	---

Description

Skew loadings L_{ij} act on the correlation scale, so they are restricted to $[-1, 1]$.

Usage

sun44_bounds()

Value

List with lower and upper length-36 numeric vectors.

See Also

[make_sun44_params\(\)](#), [fit_sun44_quad\(\)](#)

sun44_hs_bounds	<i>Default box constraints for SUN(4,4) hyperspherical parameters</i>
-----------------	---

Description

Independent-SN pair slots use $\text{atanh}(\pm 0.8)$ and all other joint coordinates use $\text{atanh}(\pm 0.5)$.

Usage

sun44_hs_bounds()

Value

List of length-36 lower and upper vectors.

sun52_hs_bounds	<i>Default box constraints for SUN(5,2) hyperspherical parameters</i>
-----------------	---

Description

Pair slots linking V_i to U_i for $i = 1, 2$ (z61, z72) use $\text{atanh}(\pm 0.8)$; other joint coordinates use $\text{atanh}(\pm 0.5)$.

Usage

sun52_hs_bounds()

Value

List of length-31 lower and upper vectors.

weighted_cov	<i>Weighted covariance</i>
--------------	----------------------------

Description

Weighted covariance

Usage

weighted_cov(X, w)

Arguments

- X Numeric matrix of observations (rows).
- w Nonnegative weights (recycled / renormalized).

Value

Symmetric ncol(X) covariance matrix.

weighted_mean	<i>Weighted column means</i>
---------------	------------------------------

Description

Weighted column means

Usage

weighted_mean(X, w)

Arguments

- X Numeric matrix of observations (rows).
- w Nonnegative weights (recycled / renormalized).

Value

Named numeric vector of length ncol(X).

Index

admvn (admvn-package), 4
admvn-package, 4

dsun, 4
dsun(), 6, 7, 9, 17
dsun22, 8
dsun22(), 37
dsun22_fun, 9
dsun22_fun(), 9, 21, 25, 37
dsun22_hs, 10
dsun22_hs(), 35
dsun22_hs_fun, 10
dsun32_hs, 11
dsun32_hs(), 38
dsun32_hs_fun, 12
dsun42_hs, 13
dsun42_hs(), 42
dsun42_hs_fun, 13
dsun43_hs, 14
dsun43_hs(), 43
dsun43_hs_fun, 15
dsun44, 16
dsun44(), 47
dsun44_fun, 17
dsun44_fun(), 17, 24, 30, 47
dsun44_hs, 18
dsun44_hs(), 45
dsun44_hs_fun, 18
dsun52_hs, 19
dsun52_hs(), 48
dsun52_hs_fun, 20
dsun_fun, 5
dsun_fun(), 5, 7, 8, 23, 28, 52
dsun_hs, 6
dsun_hs(), 8, 33, 55
dsun_hs_fun, 7
dsun_hs_fun(), 7, 22, 27

evaluate_sun22_slice, 21
evaluate_sun32_hs_slice, 21

evaluate_sun33_hs_slice, 22
evaluate_sun33_slice, 23
evaluate_sun33_slice(), 22
evaluate_sun42_hs_slice, 23
evaluate_sun43_hs_slice, 24
evaluate_sun44_slice, 24
evaluate_sun52_hs_slice, 25

fit_sun22_quad, 25
fit_sun22_quad(), 52, 53
fit_sun32_hs_quad, 26
fit_sun33_hs_quad, 27
fit_sun33_hs_quad(), 52
fit_sun33_quad, 28
fit_sun33_quad(), 27, 30, 52, 54
fit_sun42_hs_quad, 29
fit_sun43_hs_quad, 29
fit_sun44_quad, 30
fit_sun44_quad(), 52, 56
fit_sun52_hs_quad, 31

gh_points, 32

make_sun22_hs_params, 35
make_sun22_hs_params(), 10, 11, 36
make_sun22_hs_start_from_gh, 35
make_sun22_hs_start_from_normal, 36
make_sun22_params, 36
make_sun22_params(), 8, 9, 35, 38, 53
make_sun22_start_from_gh, 37
make_sun22_start_from_normal, 37
make_sun22_start_from_normal(), 26, 35, 37
make_sun32_hs_params, 38
make_sun32_hs_params(), 12, 39
make_sun32_hs_start_from_gh, 38
make_sun32_hs_start_from_normal, 39
make_sun32_hs_start_from_normal(), 27, 39
make_sun33_hs_start_from_gh, 39

make_sun33_hs_start_from_normal, 40
 make_sun33_hs_start_from_normal(), 27, 40
 make_sun33_start_from_gh, 40
 make_sun33_start_from_normal, 41
 make_sun33_start_from_normal(), 28, 41, 45, 47
 make_sun42_hs_params, 41
 make_sun42_hs_params(), 13, 14, 42, 43
 make_sun42_hs_start_from_gh, 42
 make_sun42_hs_start_from_normal, 42
 make_sun42_hs_start_from_normal(), 29, 42
 make_sun43_hs_params, 43
 make_sun43_hs_params(), 15, 44
 make_sun43_hs_start_from_gh, 44
 make_sun43_hs_start_from_normal, 44
 make_sun43_hs_start_from_normal(), 30, 44
 make_sun44_hs_params, 45
 make_sun44_hs_params(), 18, 19, 46
 make_sun44_hs_start_from_gh, 45
 make_sun44_hs_start_from_normal, 46
 make_sun44_params, 46
 make_sun44_params(), 16, 17, 45, 48, 56
 make_sun44_start_from_gh, 47
 make_sun44_start_from_normal, 47
 make_sun44_start_from_normal(), 31
 make_sun52_hs_params, 48
 make_sun52_hs_params(), 20, 49
 make_sun52_hs_start_from_gh, 48
 make_sun52_hs_start_from_normal, 49
 make_sun52_hs_start_from_normal(), 31, 48
 make_sun_hs_params, 32
 make_sun_hs_params(), 6, 7, 22, 24, 25, 40, 55
 make_sun_params, 33
 make_sun_params(), 5, 6, 32, 33, 36, 37, 41, 46, 47, 54

 pack_genz_ch, 49
 pmvn, 50
 pmvn(), 51
 pmvn_fun, 51
 pmvn_fun(), 49
 print_sun_kl_fit, 52

 sun22_bounds, 53
 sun22_bounds(), 26
 sun22_hs_bounds, 53
 sun22_hs_bounds(), 35
 sun32_hs_bounds, 54
 sun32_hs_bounds(), 27, 38
 sun33_bounds, 54
 sun33_bounds(), 28, 31, 53
 sun33_hs_bounds, 55
 sun33_hs_bounds(), 27, 33
 sun42_hs_bounds, 55
 sun42_hs_bounds(), 29, 42
 sun43_hs_bounds, 56
 sun43_hs_bounds(), 30, 43
 sun44_bounds, 56
 sun44_hs_bounds, 57
 sun44_hs_bounds(), 45
 sun52_hs_bounds, 57
 sun52_hs_bounds(), 31, 48
 sun_mle, 52
 sun_mle(), 8

 weighted_cov, 57
 weighted_mean, 58