Package ‘pharmr’

June 23, 2022

Encoding UTF-8

Version 0.73.1

Date 2022-06-22

Title Interface to the 'Pharmpy' 'Pharmacometrics' Library

Maintainer Rikard Nordgren <rikard.nordgren@farmaci.uu.se>

Depends R (>= 3.6.0), altair (>= 4.0.0)

SystemRequirements Python (>= 3.7.0)

Imports reticulate (>= 1.19), utils

Suggests testthat, magrittr, here, knitr

NeedsCompilation no

Description Interface to the 'Pharmpy' 'pharmacometrics' library. The 'Reticulate' package is used to interface Python from R.

Config/reticulate list(packages = list(list(package = ``altair''), list(package = ``pharmpy-core'')))

URL https://github.com/pharmpy/pharmr

BugReports https://github.com/pharmpy/pharmr/issues

License BSD_2_clause + file LICENSE

RoxygenNote 7.2.0

Author Rikard Nordgren [aut, cre, cph],
Stella Belin [aut, cph],
Mats O. Karlsson [sad],
Andrew C. Hooker [sad],
Sebastian Ueckert [sad] (<https://orcid.org/0000-0002-3712-0255>),
Simon Carter [rev],
Simon Buatois [rev],
João A. Abrantes [rev],
F. Hoffmann-La Roche Ltd. [fnd]

Repository CRAN

Date/Publication 2022-06-23 12:20:02 UTC

R topics documented:

add_allometry *add_allometry*

Description

Add allometric scaling of parameters

Add an allometric function to each listed parameter. The function will be P=P*(X/Z)**T where P is the parameter, X the allometric_variable, Z the reference_value and T is a theta. Default is to automatically use clearance and volume parameters.

Usage

```
add_allometry(
  model,
  allometric_variable = "WT",
  reference_value = 70,
  parameters = NULL,
  initials = NULL,
  lower_bounds = NULL,
  upper_bounds = NULL,
  fixed = TRUE
)
```

Arguments

`model`	(Model) Pharmpy model
`allometric_variable`	(str or Symbol) Variable to use for allometry (X above)
`reference_value`	(str, integer, numeric or expression) Reference value (Z above)
`parameters`	(vector) Parameters to use or NULL (default) for all available CL, Q and V parameters
`initials`	(vector) Initial estimates for the exponents. Default is to use 0.75 for CL and Qs and 1 for Vs
`lower_bounds`	(vector) Lower bounds for the exponents. Default is 0 for all parameters
`upper_bounds`	(vector) Upper bounds for the exponents. Default is 2 for all parameters
`fixed`	(logical) Whether the exponents should be fixed

Value

(Model) Pharmpy model object

Examples

```
## Not run:
model <- load_example_model("pheno")
add_allometry(model, allometric_variable='WGT')
model$statements$before_odes

## End(Not run)
```

`add_covariance_step` *add_covariance_step*

Description

Adds covariance step to the final estimation step

Usage

```
add_covariance_step(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to the same model object

See Also

add_estimation_step

set_estimation_step

remove_estimation_step

append_estimation_step_options

remove_covariance_step

set_evaluation_step

Examples

```
## Not run:
model <- load_example_model("pheno")
set_estimation_step(model, 'FOCE', cov=FALSE)
add_covariance_step(model)
ests <- model$estimation_steps
ests[1]

## End(Not run)
```

`add_covariate_effect` *add_covariate_effect*

Description

Adds covariate effect to :class:`pharmpy.model`.

The following effects have templates:

- Linear function for continuous covariates (*lin*)
- Function:

math::

coveff = 1 + theta * (cov - median)

- Init: 0.001
- Upper:
- If median of covariate equals minimum: :math:`100,000`
- Otherwise: :math:`frac{1}{{median} - {min}}`
- Lower:
- If median of covariate equals maximum: :math:`-100,000`
- Otherwise: :math:`frac{1}{{median} - {max}}`
- Linear function for categorical covariates (*cat*)
- Function:
- If covariate is most common category:

math::

coveff = 1

- For each additional category:

math::

coveff = 1 + theta

- Init: :math:`0.001`
- Upper: :math:`100,000`
- Lower: :math:`-100,000`
- Piecewise linear function/"hockey-stick", continuous covariates only (*piece_lin*)
- Function:
- If cov <= median:

math::

coveff = 1 + theta1 * (cov - median)

- If cov > median:

math::

coveff = 1 + theta2 * (cov - median)

- Init: :math:`0.001`
- Upper:
- For first state: :math:`frac{1}{{median} - {min}}`
- Otherwise: :math:`100,000`
- Lower:
- For first state: :math:`-100,000`
- Otherwise: :math:`frac{1}{{median} - {max}}`
- Exponential function, continuous covariates only (*exp*)
- Function:

math::

coveff = exp(theta * (cov - median))

- Init:
- If lower > 0.001 or upper < 0.001: :math:`frac{{upper} - {lower}}{2}`
- If estimated init is 0: :math:`frac{{upper}}{2}`
- Otherwise: :math:`0.001`
- Upper:
- If min - median = 0 or max - median = 0: :math:`100`
- Otherwise:

math::

min(fraclog(0.01)min - median, fraclog(100)max - median)

- Lower:
- If min - median = 0 or max - median = 0: :math:`0.01`
- Otherwise:

math::

max(fraclog(0.01)max - median, fraclog(100)min - median)

- Power function, continuous covariates only (*pow*)
- Function:

math::

coveff = (fraccovmedian)^theta

- Init: :math:`0.001`
- Upper: :math:`100,000`
- Lower: :math:`-100`

Usage

```
add_covariate_effect(model, parameter, covariate, effect, operation = "*")
```

Arguments

model	(Model) Pharmpy model to add covariate effect to.
parameter	(str) Name of parameter to add covariate effect to.
covariate	(str) Name of covariate.
effect	(str) Type of covariate effect. May be abbreviated covariate effect (see above) or custom.
operation	(str, optional) Whether the covariate effect should be added or multiplied (default).

Value

(Model) Reference to the same model

Examples

```
## Not run:
model <- load_example_model("pheno")
add_covariate_effect(model, "CL", "APGR", "exp")
model$statements$before_odes$full_expression("CL")

## End(Not run)
```

---

add_estimation_step *add_estimation_step*

Description

Add estimation step

Adds estimation step for a model in a given index. Methods currently supported are: FO, FOCE, ITS, LAPLACE, IMPMAP, IMP, SAEM

Usage

```
add_estimation_step(model, method, idx = NULL, ...)
```

Arguments

model	(Model) Pharmpy model
method	(str) estimation method to change to
idx	(integer) index of estimation step (starting from 0), default is NULL (adds step at the end)
...	Arguments to pass to EstimationStep (such as interaction, evaluation)

Value

(Model) Reference to the same model object

See Also

set_estimation_step

remove_estimation_step

append_estimation_step_options

add_covariance_step

remove_covariance_step

set_evaluation_step

Examples

```
## Not run:
model <- load_example_model("pheno")
opts <- list('NITER'=1000, 'ISAMPLE'=100)
add_estimation_step(model, "IMP", tool_options=opts)
ests <- model$estimation_steps
length(ests)
ests[2]

## End(Not run)
```

`add_iiv` *add_iiv*

Description

Adds IIVs to :class:`pharmpy.model`.

Effects that currently have templates are:

- Additive (*add*)
- Proportional (*prop*)
- Exponential (*exp*)
- Logit (*log*)

For all except exponential the operation input is not needed. Otherwise user specified input is supported. Initial estimates for new etas are 0.09.

Usage

```
add_iiv(
  model,
  list_of_parameters,
  expression,
  operation = "*",
  initial_estimate = 0.09,
  eta_names = NULL
)
```

Arguments

`model`	(Model) Pharmpy model to add new IIVs to.
`list_of_parameters`	(str, vector) Name/names of parameter to add new IIVs to.
`expression`	(str, vector) Effect/effects on eta. Either abbreviated (see above) or custom.
`operation`	(str, vector, optional) Whether the new IIV should be added or multiplied (default).
`initial_estimate`	(numeric) Value of initial estimate of parameter. Default is 0.09
`eta_names`	(str, vector, optional) Custom name/names of new eta

Value

(Model) Reference to the same model

See Also

add_pk_iiv

add_iov

remove_iiv

remove_iov

Examples

```
## Not run:
model <- load_example_model("pheno")
remove_iiv(model, "CL")
add_iiv(model, "CL", "add")
model$statements$find_assignment("CL")

## End(Not run)
```

add_individual_parameter *add_individual_parameter*

Description

Add an individual or pk parameter to a model

Usage

```
add_individual_parameter(model, name)
```

Arguments

`model`	(Model) Pharmpy model
`name`	(str) Name of individual/pk parameter

Value

(Model) Reference to same model

Examples

```
## Not run:
model <- load_example_model("pheno")
add_individual_parameter(model, "KA")
model$statements$find_assignment("KA")

## End(Not run)
```

add_iov *add_iov*

Description

Adds IOVs to :class:`pharmpy.model`.

Initial estimate of new IOVs are 10% of the IIV eta it is based on.

Usage

```
add_iov(
  model,
  occ,
  list_of_parameters = NULL,
  eta_names = NULL,
  distribution = "disjoint"
)
```

Arguments

`model`	(Model) Pharmpy model to add new IOVs to.
`occ`	(str) Name of occasion column.
`list_of_parameters`	(str, vector) List of names of parameters and random variables. Accepts random variable names, parameter names, or a mix of both.
`eta_names`	(str, vector) Custom names of new etas. Must be equal to the number of input etas times the number of categories for occasion.
`distribution`	(str) The distribution that should be used for the new etas. Options are 'disjoint' for disjoint normal distributions, 'joint' for joint normal distribution, 'explicit' for an explicit mix of joint and disjoint distributions, and 'same-as-iiv' for copying the distribution of IIV etas.

Value

(Model) Reference to the same model

See Also

add_iiv

add_pk_iiv

remove_iiv

remove_iov

Examples

```
## Not run:
model <- load_example_model("pheno")
add_iov(model, "TIME", "CL")
model$statements$find_assignment("CL")

## End(Not run)
```

`add_lag_time` *add_lag_time*

Description

Add lag time to the dose compartment of model.

Initial estimate for lag time is set the previous lag time if available, otherwise it is set to the time of first observation/2.

Usage

```
add_lag_time(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to same model

See Also

set_transit_compartments

remove_lag_time

Examples

```
## Not run:
model <- load_example_model("pheno")
add_lag_time(model)

## End(Not run)
```

---

add_peripheral_compartment
add_peripheral_compartment

Description

Add a peripheral distribution compartment to model

The rate of flow from the central to the peripheral compartment will be parameterized as QPn / VC where VC is the volume of the central compartment. The rate of flow from the peripheral to the central compartment will be parameterized as QPn / VPn where VPn is the volumne of the added peripheral compartment.

Initial estimates:

== === n == ================================
1 :math:`{CL} = {CL'}`, :math:`{VC} = {VC'}`, :math:`{QP1} = {CL'}` and :math:`{VP1} = {VC'} * 0.05`
2 :math:`{QP1} = {QP1' / 2}`, :math:`{VP1} = {VP1'}`, :math:`{QP2} = {QP1' / 2}` and :math:`{VP2} = {VP1'}` == ===

Usage

```
add_peripheral_compartment(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to same model

See Also

set_peripheral_compartment

remove_peripheral_compartment

Examples

```
## Not run:
model <- load_example_model("pheno")
add_peripheral_compartment(model)
model$statements$ode_system

## End(Not run)
```

`add_pk_iiv` *add_pk_iiv*

Description

Adds IIVs to all PK parameters in :class:`pharmpy.model`.

Will add exponential IIVs to all parameters that are included in the ODE.

Usage

```
add_pk_iiv(model, initial_estimate = 0.09)
```

Arguments

`model`	(Model) Pharmpy model to add new IIVs to.
`initial_estimate`	(numeric) Value of initial estimate of parameter. Default is 0.09

Value

(Model) Reference to the same model

See Also

add_iiv

add_iov

remove_iiv

remove_iov

Examples

```
## Not run:
model <- load_example_model("pheno")
set_first_order_absorption(model)
model$statements$find_assignment("MAT")
add_pk_iiv(model)
model$statements$find_assignment("MAT")

## End(Not run)
```

`add_population_parameter` *add_population_parameter*

Description

Add a new population parameter to the model

Usage

```
add_population_parameter(
  model,
  name,
  init,
  lower = NULL,
  upper = NULL,
  fix = FALSE
)
```

Arguments

`model`	(Model) Pharmpy model
`name`	(str) Name of the new parameter
`init`	(numeric) Initial estimate of the new parameter
`lower`	(numeric) Lower bound of the new parameter
`upper`	(numeric) Upper bound of the new parameter
`fix`	(logical) Should the new parameter be fixed?

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
add_population_parameter(model, 'POP_KA', 2)
model$parameters

## End(Not run)
```

add_time_after_dose *add_time_after_dose*

Description

Calculate and add a TAD column to the dataset"

Usage

```
add_time_after_dose(model)
```

Arguments

model	(Model) Pharmpy model

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
add_time_after_dose(model)

## End(Not run)
```

append_estimation_step_options
append_estimation_step_options

Description

Append estimation step options

Appends options to an existing estimation step.

Usage

```
append_estimation_step_options(model, tool_options, idx)
```

Arguments

`model`	(Model) Pharmpy model
`tool_options`	(list) any additional tool specific options
`idx`	(integer) index of estimation step (starting from 0)

Value

(Model) Reference to the same model object

See Also

add_estimation_step

set_estimation_step

remove_estimation_step

add_covariance_step

remove_covariance_step

set_evaluation_step

Examples

```
## Not run:
model <- load_example_model("pheno")
opts <- list('NITER'=1000, 'ISAMPLE'=100)
append_estimation_step_options(model, tool_options=opts, idx=0)
est <- model$estimation_steps[1]
length(est$tool_options)

## End(Not run)
```

bump_model_number *bump_model_number*

Description

If the model name ends in a number increase it

If path is set increase the number until no file exists with the same name in path. If model name does not end in a number do nothing.

Usage

```
bump_model_number(model, path = NULL)
```

Arguments

`model`	(Model) Pharmpy model object
`path`	(Path in which to find next unique number) Default is to not look for files.

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
model$name <- "run2"
bump_model_number(model)
model$name

## End(Not run)
```

calculate_aic *calculate_aic*

Description

Calculate final AIC for model assuming the OFV to be -2LL

AIC = OFV + 2*n_estimated_parameters

Usage

```
calculate_aic(model, modelfit_results = NULL)
```

Arguments

`model`	(Model) Pharmpy model object
`modelfit_results`	(ModelfitResults) Alternative results object. Default is to use the one in model

Value

(numeric) AIC of model fit

`calculate_bic` *calculate_bic*

Description

Calculate final BIC value assuming the OFV to be -2LL

Different variations of the BIC can be calculated:

- | mixed (default) | BIC = OFV + n_random_parameters * log(n_individuals) + | n_fixed_parameters * log(n_observations)
- | fixed | BIC = OFV + n_estimated_parameters * log(n_observations)
- | random | BIC = OFV + n_estimated_parameters * log(n_individals)
- | iiv | BIC = OFV + n_estimated_iiv_omega_parameters * log(n_individals)

Usage

```
calculate_bic(model, type = NULL, modelfit_results = NULL)
```

Arguments

`model`	(Model) Pharmpy model object
`type`	(str) Type of BIC to calculate. Default is the mixed effects.
`modelfit_results`	(ModelfitResults) Alternative results object. Default is to use the one in model

Value

(numeric) BIC of model fit

Examples

```
## Not run:
model <- load_example_model("pheno")
calculate_bic(model)
calculate_bic(model, type='fixed')
calculate_bic(model, type='random')
calculate_bic(model, type='iiv')

## End(Not run)
```

calculate_corr_from_cov

calculate_corr_from_cov

Description

Calculate correlation matrix from a covariance matrix

Usage

```
calculate_corr_from_cov(cov)
```

Arguments

cov	(data.frame) Covariance matrix

Value

(data.frame) Correlation matrix

See Also

calculate_se_from_cov : Standard errors from covariance matrix

calculate_se_from_inf : Standard errors from information matrix

calculate_cov_from_inf : Covariance matrix from information matrix

calculate_cov_from_corrse : Covariance matrix from correlation matrix and standard errors

calculate_inf_from_cov : Information matrix from covariance matrix

calculate_inf_from_corrse : Information matrix from correlation matrix and standard errors

calculate_corr_from_inf : Correlation matrix from information matrix

Examples

```
## Not run:
model <- load_example_model("pheno")
cov <- model$modelfit_results$covariance_matrix
cov
calculate_corr_from_cov(cov)

## End(Not run)
```

`calculate_corr_from_inf`
calculate_corr_from_inf

Description

Calculate correlation matrix from an information matrix

Usage

```
calculate_corr_from_inf(information_matrix)
```

Arguments

`information_matrix`	(data.frame) Information matrix

Value

(data.frame) Correlation matrix

See Also

calculate_se_from_cov : Standard errors from covariance matrix

calculate_se_from_inf : Standard errors from information matrix

calculate_corr_from_cov : Correlation matrix from covariance matrix

calculate_cov_from_inf : Covariance matrix from information matrix

calculate_cov_from_corrse : Covariance matrix from correlation matrix and standard errors

calculate_inf_from_cov : Information matrix from covariance matrix

calculate_inf_from_corrse : Information matrix from correlation matrix and standard errors

Examples

```
## Not run:
model <- load_example_model("pheno")
inf <- model$modelfit_results$information_matrix
inf
calculate_corr_from_inf(inf)

## End(Not run)
```

calculate_cov_from_corrse

calculate_cov_from_corrse

Description

Calculate covariance matrix from a correlation matrix and standard errors

Usage

```
calculate_cov_from_corrse(corr, se)
```

Arguments

`corr`	(data.frame) Correlation matrix
`se`	(data.frame) Standard errors

Value

(data.frame) Covariance matrix

See Also

calculate_se_from_cov : Standard errors from covariance matrix

calculate_se_from_inf : Standard errors from information matrix

calculate_corr_from_cov : Correlation matrix from covariance matrix

calculate_cov_from_inf : Covariance matrix from information matrix

calculate_inf_from_cov : Information matrix from covariance matrix

calculate_inf_from_corrse : Information matrix from correlation matrix and standard errors

calculate_corr_from_inf : Correlation matrix from information matrix

Examples

```
## Not run:
model <- load_example_model("pheno")
corr <- model$modelfit_results$correlation_matrix
se <- model$modelfit_results$standard_errors
corr
calculate_cov_from_corrse(corr, se)

## End(Not run)
```

calculate_cov_from_inf *calculate_cov_from_inf*

Description

Calculate covariance matrix from an information matrix

Usage

```
calculate_cov_from_inf(information_matrix)
```

Arguments

`information_matrix`	(data.frame) Information matrix

Value

(data.frame) Covariance matrix

See Also

calculate_se_from_cov : Standard errors from covariance matrix

calculate_se_from_inf : Standard errors from information matrix

calculate_corr_from_cov : Correlation matrix from covariance matrix

calculate_cov_from_corrse : Covariance matrix from correlation matrix and standard errors

calculate_inf_from_cov : Information matrix from covariance matrix

calculate_inf_from_corrse : Information matrix from correlation matrix and standard errors

calculate_corr_from_inf : Correlation matrix from information matrix

Examples

```
## Not run:
model <- load_example_model("pheno")
inf <- model$modelfit_results$information_matrix
inf
calculate_cov_from_inf(inf)

## End(Not run)
```

calculate_epsilon_gradient_expression

calculate_epsilon_gradient_expression

Description

Calculate the symbolic expression for the epsilon gradient

This function currently only support models without ODE systems

Usage

```
calculate_epsilon_gradient_expression(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Expression) Symbolic expression

See Also

calculate_eta_gradient_expression : Eta gradient

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
calculate_epsilon_gradient_expression(model)

## End(Not run)
```

calculate_eta_gradient_expression

calculate_eta_gradient_expression

Description

Calculate the symbolic expression for the eta gradient

This function currently only support models without ODE systems

Usage

```
calculate_eta_gradient_expression(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Expression) Symbolic expression

See Also

calculate_epsilon_gradient_expression : Epsilon gradient

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
calculate_eta_gradient_expression(model)

## End(Not run)
```

`calculate_eta_shrinkage` *calculate_eta_shrinkage*

Description

Calculate eta shrinkage for each eta

Usage

```
calculate_eta_shrinkage(model, sd = FALSE)
```

Arguments

`model`	(Model) Pharmpy model
`sd`	(logical) Calculate shrinkage on the standard deviation scale (default is to calculate on the variance scale)

Value

(Series) Shrinkage for each eta

See Also

calculate_individual_shrinkage

Examples

```
## Not run:
model <- load_example_model("pheno")
calculate_eta_shrinkage(model)
calculate_eta_shrinkage(model, sd=TRUE)

## End(Not run)
```

---

calculate_individual_parameter_statistics

calculate_individual_parameter_statistics

Description

Calculate statistics for individual parameters

Calculate the mean (expected value of the distribution), variance (variance of the distribution) and standard error for individual parameters described by arbitrary expressions. Any dataset column or variable used in the model can be used in the expression. The exception being that variables that depends on the solution of the ODE system cannot be used. If covariates are used in the expression the statistics of the parameter is calculated at the median value of each covariate as well as at the 5:th and 95:th percentiles. If no parameter uncertainty is available for the model the standard error will not be calculated.

Usage

```
calculate_individual_parameter_statistics(model, exprs, rng = NULL)
```

Arguments

model	(Model) A previously estimated model
exprs	(str, sympy expression or iterable of str or sympy expressions) Expressions or equations for parameters of interest. If equations are used the names of the left hand sides will be used as the names of the parameters.
rng	(Generator or integer) Random number generator or integer seed

Value

(data.frame) A DataFrame of statistics indexed on parameter and covariate value.

Examples

```
## Not run:
model <- load_example_model("pheno")
rng <- create_rng(23)
calculate_individual_parameter_statistics(model, "K=CL/V", rng=rng)

## End(Not run)
```

calculate_individual_shrinkage *calculate_individual_shrinkage*

Description

Calculate the individual eta-shrinkage

Definition: ieta_shr = (var(eta) / omega)

Usage

```
calculate_individual_shrinkage(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(DataFrame) Shrinkage for each eta and individual

See Also

calculate_eta_shrinkage

Examples

```
## Not run:
model <- load_example_model("pheno")
calculate_individual_shrinkage(model)

## End(Not run)
```

calculate_inf_from_corrse *calculate_inf_from_corrse*

Description

Calculate information matrix from a correlation matrix and standard errors

Usage

```
calculate_inf_from_corrse(corr, se)
```

Arguments

`corr`	(data.frame) Correlation matrix
`se`	(data.frame) Standard errors

Value

(data.frame) Information matrix

See Also

calculate_se_from_cov : Standard errors from covariance matrix

calculate_se_from_inf : Standard errors from information matrix

calculate_corr_from_cov : Correlation matrix from covariance matrix

calculate_cov_from_inf : Covariance matrix from information matrix

calculate_cov_from_corrse : Covariance matrix from correlation matrix and standard errors

calculate_inf_from_cov : Information matrix from covariance matrix

calculate_corr_from_inf : Correlation matrix from information matrix

Examples

```
## Not run:
model <- load_example_model("pheno")
corr <- model$modelfit_results$correlation_matrix
se <- model$modelfit_results$standard_errors
corr
calculate_inf_from_corrse(corr, se)

## End(Not run)
```

`calculate_inf_from_cov` *calculate_inf_from_cov*

Description

Calculate information matrix from a covariance matrix

Usage

```
calculate_inf_from_cov(cov)
```

Arguments

`cov`	(data.frame) Covariance matrix

Value

(data.frame) Information matrix

See Also

calculate_se_from_cov : Standard errors from covariance matrix

calculate_se_from_inf : Standard errors from information matrix

calculate_corr_from_cov : Correlation matrix from covariance matrix

calculate_cov_from_inf : Covariance matrix from information matrix

calculate_cov_from_corrse : Covariance matrix from correlation matrix and standard errors

calculate_inf_from_corrse : Information matrix from correlation matrix and standard errors

calculate_corr_from_inf : Correlation matrix from information matrix

Examples

```
## Not run:
model <- load_example_model("pheno")
cov <- model$modelfit_results$covariance_matrix
cov
calculate_inf_from_cov(cov)

## End(Not run)
```

---

calculate_parameters_from_ucp

calculate_parameters_from_ucp

Description

Scale parameter values from ucp to normal scale

Usage

```
calculate_parameters_from_ucp(model, scale, ucps)
```

Arguments

`model`	(Model) Pharmpy model
`scale`	(UCPSCale) A parameter scale
`ucps`	(data.frame or list) Series of parameter values

Value

(data.frame) Parameters on the normal scale

See Also

calculate_ucp_scale : Calculate the scale for conversion from ucps

Examples

```
## Not run:
model <- load_example_model("pheno")
scale <- calculate_ucp_scale(model)
values <- list('THETA(1)'=0.1, 'THETA(2)'=0.1, 'THETA(3)'=0.1,
               'OMEGA(1,1)'=0.1, 'OMEGA(2,2)'=0.1, 'SIGMA(1,1)'=0.1)
calculate_parameters_from_ucp(model, scale, values)

## End(Not run)
```

calculate_pk_parameters_statistics
calculate_pk_parameters_statistics

Description

Calculate statistics for common pharmacokinetic parameters

Calculate the mean (expected value of the distribution), variance (variance of the distribution) and standard error for some individual pre-defined pharmacokinetic parameters.

Usage

```
calculate_pk_parameters_statistics(model, rng = NULL)
```

Arguments

model	(Model) A previously estimated model
rng	(Generator or integer) Random number generator or seed

Value

(data.frame) A DataFrame of statistics indexed on parameter and covariate value.

See Also

calculate_individual_parameter_statistics : Calculation of statistics for arbitrary parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
rng <- create_rng(23)
calculate_pk_parameters_statistics(model, rng=rng)

## End(Not run)
```

`calculate_se_from_cov` *calculate_se_from_cov*

Description

Calculate standard errors from a covariance matrix

Usage

```
calculate_se_from_cov(cov)
```

Arguments

`cov`	(data.frame) Input covariance matrix

Value

(data.frame) Standard errors

See Also

calculate_se_from_inf : Standard errors from information matrix

calculate_corr_from_cov : Correlation matrix from covariance matrix

calculate_cov_from_inf : Covariance matrix from information matrix

calculate_cov_from_corrse : Covariance matrix from correlation matrix and standard errors

calculate_inf_from_cov : Information matrix from covariance matrix

calculate_inf_from_corrse : Information matrix from correlation matrix and standard errors

calculate_corr_from_inf : Correlation matrix from information matrix

Examples

```
## Not run:
model <- load_example_model("pheno")
cov <- model$modelfit_results$covariance_matrix
cov
calculate_se_from_cov(cov)

## End(Not run)
```

`calculate_se_from_inf` *calculate_se_from_inf*

Description

Calculate standard errors from an information matrix

Usage

```
calculate_se_from_inf(information_matrix)
```

Arguments

`information_matrix`	(data.frame) Input information matrix

Value

(data.frame) Standard errors

See Also

calculate_se_from_cov : Standard errors from covariance matrix

calculate_corr_from_cov : Correlation matrix from covariance matrix

calculate_cov_from_inf : Covariance matrix from information matrix

calculate_cov_from_corrse : Covariance matrix from correlation matrix and standard errors

calculate_inf_from_cov : Information matrix from covariance matrix

calculate_inf_from_corrse : Information matrix from correlation matrix and standard errors

calculate_corr_from_inf : Correlation matrix from information matrix

Examples

```
## Not run:
model <- load_example_model("pheno")
inf <- model$modelfit_results$information_matrix
inf
calculate_se_from_inf(inf)

## End(Not run)
```

`calculate_ucp_scale` *calculate_ucp_scale*

Description

Calculate a scale for unconstrained parameters for a model

The UCPScale object can be used to calculate unconstrained parameters back into the normal parameter space.

Usage

```
calculate_ucp_scale(model)
```

Arguments

`model`	(Model) Model for which to calculate an ucp scale

Value

(UCPScale) A scale object

See Also

calculate_parameters_from_ucp : Calculate parameters from ucp:s

Examples

```
## Not run:
model <- load_example_model("pheno")
scale <- calculate_ucp_scale(model)

## End(Not run)
```

`check_dataset` *check_dataset*

Description

Check dataset for consistency across a set of rules

Usage

```
check_dataset(model, dataframe = FALSE, verbose = FALSE)
```

Arguments

`model`	(Model) Pharmpy model object
`dataframe`	(Bool) TRUE to return a DataFrame instead of printing to the console
`verbose`	(Bool) Print out all rules checked if TRUE else print only failed rules

Value

(data.frame) Only returns a DataFrame is dataframe=TRUE

`check_high_correlations`

check_high_correlations

Description

Check for highly correlated parameter estimates

Usage

```
check_high_correlations(model, limit = 0.9)
```

Arguments

`model`	(Model) Pharmpy model object
`limit`	(numeric) Lower limit for a high correlation

Value

(data.frame) Correlation values indexed on pairs of parameters for (absolute) correlations above limit

Examples

```
## Not run:
model <- load_example_model("pheno")
check_high_correlations(model, limit=0.3)

## End(Not run)
```

check_parameters_near_bounds
check_parameters_near_bounds

Description

Check if any estimated parameter value is close to its bounds

Usage

```
check_parameters_near_bounds(
  model,
  values = NULL,
  zero_limit = 0.001,
  significant_digits = 2
)
```

Arguments

model	(Model) Pharmpy model object
values	(data.frame) Series of values with index a subset of parameter names. Default is to use all parameter estimates
zero_limit	(number) maximum distance to 0 bounds
significant_digits	(integer) maximum distance to non-zero bounds in number of significant digits

Value

(data.frame) Logical Series with same index as values

Examples

```
## Not run:
model <- load_example_model("pheno")
check_parameters_near_bounds(model)

## End(Not run)
```

`check_pharmpy` *Checks version of Pharmpy/pharmr*

Description

Checks whether Pharmpy and pharmr has the same version

Usage

```
check_pharmpy(pharmpy_version)
```

Arguments

`pharmpy_version`	(str) version number as string

`cleanup_model` *cleanup_model*

Description

Perform various cleanups of a model

This is what is currently done

- Make model statements declarative, i.e. only one assignment per symbol
- Inline all assignments of one symbol, e.g. X = Y

Usage

```
cleanup_model(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to the same model

Note

When creating NONMEM code from the cleaned model Pharmpy might need toadd certain assignments to make it in line with what NONMEM requires.

Examples

```
## Not run:
model <- load_example_model("pheno")
model$statements
cleanup_model(model)
model$statements

## End(Not run)
```

---

convert_model *convert_model*

Description

Convert model to other format

Note that the operation is not done inplace.

Usage

```
convert_model(model, to_format)
```

Arguments

`model`	(Model) Model to convert
`to_format`	(str) Name of format to convert into. Currently supported 'generic', 'nlmixr' and 'nonmem'

Value

(Model) New model object with new underlying model format

Examples

```
## Not run:
model <- load_example_model("pheno")
converted_model <- convert_model(model, "nlmixr")

## End(Not run)
```

copy_model *copy_model*

Description

Copies model to a new model object

Usage

```
copy_model(model, name = NULL)
```

Arguments

model	(Model) Pharmpy model
name	(str) Optional new name of model

Value

(Model) A copy of the input model

Examples

```
## Not run:
model <- load_example_model("pheno")
model_copy <- copy_model(model, "pheno2")

## End(Not run)
```

create_joint_distribution *create_joint_distribution*

Description

Combines some or all etas into a joint distribution.

The etas must be IIVs and cannot be fixed. Initial estimates for covariance between the etas is dependent on whether the model has results from a previous results. In that case, the correlation will be calculated from individual estimates, otherwise correlation will be set to 10%.

Usage

```
create_joint_distribution(model, rvs = NULL)
```

Arguments

`model`	(Model) Pharmpy model
`rvs`	(vector) Sequence of etas or names of etas to combine. If NULL, all etas that are IIVs and non-fixed will be used (full block). NULL is default.

Value

(Model) Reference to the same model

See Also

split_joint_distribution : split etas into separate distributions

Examples

```
## Not run:
model <- load_example_model("pheno")
model$random_variables$etas
create_joint_distribution(model, c('ETA(1)', 'ETA(2)'))
model$random_variables$etas

## End(Not run)
```

`create_report` *create_report*

Description

Create standard report for results

The report will be an html created at specified path.

Usage

```
create_report(results, path)
```

Arguments

`results`	(Results) Results for which to create report
`path`	(Path) Path to report file

`create_results` *create_results*

Description

Create/recalculate results object given path to run directory

Usage

```
create_results(path, ...)
```

Arguments

`path`	(str, Path) Path to run directory
`...`	Arguments to pass to tool specific create results function

Value

(Results) Results object for tool

See Also

read_results

Examples

```
## Not run:
res <- create_results("frem_dir1")

## End(Not run)
```

`create_rng` *create_rng*

Description

Create a new random number generator

Pharmpy functions that use random sampling take a random number generator or seed as input. This function can be used to create a default new random number generator.

Usage

```
create_rng(seed = NULL)
```

Arguments

`seed`	(integer or rng) Seed for the random number generator or NULL (default) for a randomized seed. If seed is generator it will be passed through.

Value

(Generator) Initialized numpy random number generator object

Examples

```
## Not run:
rng <- create_rng(23)
rng$standard_normal()

## End(Not run)
```

`create_symbol` *create_symbol*

Description

Create a new unique variable symbol given a model

Usage

```
create_symbol(model, stem, force_numbering = FALSE)
```

Arguments

`model`	(Model) Pharmpy model object
`stem`	(str) First part of the new variable name
`force_numbering`	(logical) Forces addition of number to name even if variable does not exist, e.g. COVEFF –> COVEFF1

Value

(Symbol) Created symbol with unique name

Examples

```
## Not run:
model <- load_example_model("pheno")
create_symbol(model, "TEMP")
create_symbol(model, "TEMP", force_numbering=TRUE)
create_symbol(model, "CL")

## End(Not run)
```

`drop_columns` *drop_columns*

Description

Drop columns from the dataset or mark as dropped

Usage

```
drop_columns(model, column_names, mark = FALSE)
```

Arguments

`model`	(Model) Pharmpy model object
`column_names`	(vector or str) List of column names or one column name to drop or mark as dropped
`mark`	(logical) Default is to remove column from dataset set this to TRUE to only mark as dropped

Value

(Model) Reference to same model object

See Also

drop_dropped_columns : Drop all columns marked as drop

undrop_columns : Undrop columns of model

Examples

```
## Not run:
model <- load_example_model("pheno")
drop_columns(model, c('WGT', 'APGR'))
vector(model$dataset$columns)

## End(Not run)
```

`drop_dropped_columns` *drop_dropped_columns*

Description

Drop columns marked as dropped from the dataset

NM-TRAN date columns will not be dropped by this function even if marked as dropped. Columns not specified in the datainfo ($INPUT for NONMEM) will also be dropped from the dataset.

Usage

```
drop_dropped_columns(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Model) Reference to same model object

See Also

drop_columns : Drop specific columns or mark them as drop

Examples

```
## Not run:
model <- load_example_model("pheno")
drop_dropped_columns(model)
vector(model$dataset$columns)

## End(Not run)
```

`evaluate_epsilon_gradient` *evaluate_epsilon_gradient*

Description

Evaluate the numeric epsilon gradient

The gradient is evaluated at the current model parameter values or optionally at the given parameter values. The gradient is done for each data record in the model dataset or optionally using the dataset argument. The gradient is done at the current eta values or optionally at the given eta values.

This function currently only support models without ODE systems

Usage

```
evaluate_epsilon_gradient(
  model,
  etas = NULL,
  parameters = NULL,
  dataset = NULL
)
```

Arguments

`model`	(Model) Pharmpy model
`etas`	(list) Optional list of eta values
`parameters`	(list) Optional list of parameters and values
`dataset`	(data.frame) Optional dataset

Value

(data.frame) Gradient

See Also

evaluate_eta_gradient : Evaluate the eta gradient

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
evaluate_epsilon_gradient(model)

## End(Not run)
```

`evaluate_eta_gradient` *evaluate_eta_gradient*

Description

Evaluate the numeric eta gradient

The gradient is evaluated at the current model parameter values or optionally at the given parameter values. The gradient is done for each data record in the model dataset or optionally using the dataset argument. The gradient is done at the current eta values or optionally at the given eta values.

This function currently only support models without ODE systems

Usage

```
evaluate_eta_gradient(model, etas = NULL, parameters = NULL, dataset = NULL)
```

Arguments

`model`	(Model) Pharmpy model
`etas`	(list) Optional list of eta values
`parameters`	(list) Optional list of parameters and values
`dataset`	(data.frame) Optional dataset

Value

(data.frame) Gradient

See Also

evaluate_epsilon_gradient : Evaluate the epsilon gradient

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
evaluate_eta_gradient(model)

## End(Not run)
```

`evaluate_expression` *evaluate_expression*

Description

Evaluate expression using model

Calculate the value of expression for each data record. The expression can contain dataset columns, variables in model and population parameters. If the model has parameter estimates these will be used. Initial estimates will be used for non-estimated parameters.

Usage

```
evaluate_expression(model, expression)
```

Arguments

`model`	(Model) Pharmpy model
`expression`	(str or sympy expression) Expression to evaluate

Value

(data.frame) A series of one evaluated value for each data record

Examples

```
## Not run:
model <- load_example_model("pheno")
evaluate_expression(model, "TVCL*1000")

## End(Not run)
```

---

evaluate_individual_prediction

evaluate_individual_prediction

Description

Evaluate the numeric individual prediction

The prediction is evaluated at the current model parameter values or optionally at the given parameter values. The evaluation is done for each data record in the model dataset or optionally using the dataset argument. The evaluation is done at the current eta values or optionally at the given eta values.

This function currently only support models without ODE systems

Usage

```
evaluate_individual_prediction(
  model,
  etas = NULL,
  parameters = NULL,
  dataset = NULL
)
```

Arguments

`model`	(Model) Pharmpy model
`etas`	(list) Optional list of eta values
`parameters`	(list) Optional list of parameters and values
`dataset`	(data.frame) Optional dataset

Value

(data.frame) Individual predictions

See Also

evaluate_population_prediction : Evaluate the population prediction

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
evaluate_individual_prediction(model)

## End(Not run)
```

---

evaluate_population_prediction

evaluate_population_prediction

Description

Evaluate the numeric population prediction

The prediction is evaluated at the current model parameter values or optionally at the given parameter values. The evaluation is done for each data record in the model dataset or optionally using the dataset argument.

This function currently only support models without ODE systems

Usage

```
evaluate_population_prediction(model, parameters = NULL, dataset = NULL)
```

Arguments

`model`	(Model) Pharmpy model
`parameters`	(list) Optional list of parameters and values
`dataset`	(data.frame) Optional dataset

Value

(data.frame) Population predictions

See Also

evaluate_individual_prediction : Evaluate the individual prediction

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
evaluate_population_prediction(model)

## End(Not run)
```

evaluate_weighted_residuals

evaluate_weighted_residuals

Description

Evaluate the weighted residuals

The residuals is evaluated at the current model parameter values or optionally at the given parameter values. The residuals is done for each data record in the model dataset or optionally using the dataset argument.

This function currently only support models without ODE systems

Usage

```
evaluate_weighted_residuals(model, parameters = NULL, dataset = NULL)
```

Arguments

`model`	(Model) Pharmpy model
`parameters`	(list) Optional list of parameters and values
`dataset`	(data.frame) Optional dataset

Value

(data.frame) WRES

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
evaluate_weighted_residuals(model)

## End(Not run)
```

expand_additional_doses

expand_additional_doses

Description

Expand additional doses into separate dose records

Usage

```
expand_additional_doses(model, flag = FALSE)
```

Arguments

`model`	(Model) Pharmpy model object
`flag`	(logical) TRUE to add a boolean EXPANDED column to mark added records. In this case all columns in the original dataset will be kept. Care needs to be taken to handle the new dataset.

Value

(Model) Reference to the same model object

`find_clearance_parameters` *find_clearance_parameters*

Description

Find clearance parameters in model

Usage

```
find_clearance_parameters(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(vector) A vector of clearance parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
find_clearance_parameters(model)

## End(Not run)
```

`find_volume_parameters` *find_volume_parameters*

Description

Find volume parameters in model

Usage

```
find_volume_parameters(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(vector) A vector of volume parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
find_volume_parameters(model)

## End(Not run)
```

`fit` *fit*

Description

Fit models.

Usage

```
fit(models, tool = NULL)
```

Arguments

`models`	(vector) List of models or one single model
`tool`	(str) Estimation tool to use. NULL to use default

Value

(Model) Reference to same model

See Also

run_tool

Examples

```
## Not run:
model <- load_example_model("pheno")
fit(model)

## End(Not run)
```

`fix_or_unfix_parameters`
fix_or_unfix_parameters

Description

Fix or unfix parameters

Set fixedness of parameters to specified values

Usage

```
fix_or_unfix_parameters(model, parameters)
```

Arguments

`model`	(Model) Pharmpy model
`parameters`	(list) Set fix/unfix for these parameters

Value

(Model) Reference to the same model object

See Also

fix_parameters : Fix parameters

unfix_paramaters : Unfixing parameters

fix_paramaters_to : Fixing parameters and setting a new initial estimate in the same function

unfix_paramaters_to : Unfixing parameters and setting a new initial estimate in the same function

Examples

```
## Not run:
model <- load_example_model("pheno")
model$parameters['THETA(1)']
fix_or_unfix_parameters(model, list('THETA(1)'=TRUE))
model$parameters['THETA(1)']

## End(Not run)
```

---

fix_parameters *fix_parameters*

Description

Fix parameters

Fix all listed parameters

Usage

```
fix_parameters(model, parameter_names)
```

Arguments

model	(Model) Pharmpy model
parameter_names	(vector or str) one parameter name or a vector of parameter names

Value

(Model) Reference to the same model object

See Also

fix_or_unfix_parameters : Fix or unfix parameters (given boolean)

fix_parameters_to : Fixing and setting parameter initial estimates in the same function

unfix_paramaters : Unfixing parameters

unfix_paramaters_to : Unfixing parameters and setting a new initial estimate in the same function

Examples

```
## Not run:
model <- load_example_model("pheno")
model$parameters['THETA(1)']
fix_parameters(model, 'THETA(1)')
model$parameters['THETA(1)']

## End(Not run)
```

`fix_parameters_to` *fix_parameters_to*

Description

Fix parameters to

Fix all listed parameters to specified value/values

Usage

```
fix_parameters_to(model, inits)
```

Arguments

`model`	(Model) Pharmpy model
`inits`	(list) Inits for all parameters to fix and set init

Value

(Model) Reference to the same model object

See Also

fix_parameters : Fix parameters

fix_or_unfix_parameters : Fix or unfix parameters (given boolean)

unfix_paramaters : Unfixing parameters

unfix_paramaters_to : Unfixing parameters and setting a new initial estimate in the same

function

Examples

```
## Not run:
model <- load_example_model("pheno")
model$parameters['THETA(1)']
fix_parameters_to(model, {'THETA(1)': 0.5})
model$parameters['THETA(1)']

## End(Not run)
```

generate_model_code *generate_model_code*

Description

Get the model code of the underlying model language

Usage

```
generate_model_code(model)
```

Arguments

model	(Model) Pharmpy model

Value

(str) Model code

Examples

```
## Not run:
model <- load_example_model("pheno")
generate_model_code(model)

## End(Not run)
```

get_baselines *get_baselines*

Description

Baselines for each subject.

Baseline is taken to be the first row even if that has a missing value.

Usage

```
get_baselines(model)
```

Arguments

model	(Model) Pharmpy model

Value

(data.frame) Dataset with the baselines

Examples

```
## Not run:
model <- load_example_model("pheno")
get_baselines(model)

## End(Not run)
```

`get_bioavailability` *get_bioavailability*

Description

Get bioavailability of doses for all compartments

Usage

```
get_bioavailability(model)
```

Arguments

`model`	(Pharmpy model) Result list Dictionary from compartment name to bioavailability expression

---

`get_concentration_parameters_from_data` *get_concentration_parameters_from_data*

Description

Create a dataframe with concentration parameters

Note that all values are directly calculated from the dataset

Usage

```
get_concentration_parameters_from_data(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(data.frame) Concentration parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
get_concentration_parameters_from_data(model)

## End(Not run)
```

`get_config_path` *get_config_path*

Description

Returns path to the user config path

Usage

```
get_config_path()
```

Value

(str) Path to user config

Examples

```
## Not run:
get_config_path()

## End(Not run)
```

`get_covariate_baselines` *get_covariate_baselines*

Description

Return a dataframe with baselines of all covariates for each id.

Baseline is taken to be the first row even if that has a missing value.

Usage

```
get_covariate_baselines(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(data.frame) covariate baselines

See Also

get_baselines : baselines for all data columns

Examples

```
## Not run:
model <- load_example_model("pheno")
model$datainfo[["WGT", "APGR"]].types <- "covariate"
get_covariate_baselines(model)

## End(Not run)
```

---

get_doseid *get_doseid*

Description

Get a DOSEID series from the dataset with an id of each dose period starting from 1

If a a dose and observation exist at the same time point the observation will be counted towards the previous dose.

Usage

```
get_doseid(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(data.frame) DOSEIDs

Examples

```
## Not run:
model <- load_example_model("pheno")
get_doseid(model)

## End(Not run)
```

get_doses *get_doses*

Description

Get a series of all doses

Indexed with ID and TIME

Usage

```
get_doses(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(data.frame) doses

Examples

```
## Not run:
model <- load_example_model("pheno")
get_doses(model)

## End(Not run)
```

get_ids *get_ids*

Description

Retrieve a vector of all subject ids of the dataset

Usage

```
get_ids(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(vector) All subject ids

Examples

```
## Not run:
model <- load_example_model("pheno")
get_ids(model)

## End(Not run)
```

---

get_individual_parameters

get_individual_parameters

Description

Retrieves all parameters with IIV or IOV in :class:`pharmpy.model`.

Usage

```
get_individual_parameters(model, level = "all")
```

Arguments

`model`	(Model) Pharmpy model to retrieve the individuals parameters from
`level`	(str) The variability level to look for: 'iiv', 'iov', or 'all' (default)

Value

(vector) A vector of the parameters' names as strings

See Also

get_pk_parameters

has_random_effect

Examples

```
## Not run:
model <- load_example_model("pheno")
sorted(get_individual_parameters(model))
sorted(get_individual_parameters(model, 'iiv'))
get_individual_parameters(model, 'iov')

## End(Not run)
```

get_individual_prediction_expression *get_individual_prediction_expression*

Description

Get the full symbolic expression for the modelled individual prediction

This function currently only support models without ODE systems

Usage

```
get_individual_prediction_expression(model)
```

Arguments

model	(Model) Pharmpy model object

Value

(Expression) Symbolic expression

See Also

get_population_prediction_expression : Get full symbolic epression for the population prediction

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
get_individual_prediction_expression(model)

## End(Not run)
```

get_lag_times *get_lag_times*

Description

Get lag times for all compartments

Usage

```
get_lag_times(model)
```

Arguments

model	(Pharmpy model) Result list Dictionary from compartment name to lag time expression

`get_mdv` *get_mdv*

Description

Get MDVs from dataset

Usage

```
get_mdv(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(data.frame) MDVs

`get_model_covariates` *get_model_covariates*

Description

List of covariates used in model

A covariate in the model is here defined to be a data item affecting the model prediction excluding dosing items.

Usage

```
get_model_covariates(model, strings = FALSE)
```

Arguments

`model`	(Model) Pharmpy model
`strings`	(logical) Return strings instead of symbols? FALSE (default) will give symbols

Value

(vector) Covariate symbols or names

Examples

```
## Not run:
model <- load_example_model("pheno")
get_model_covariates(model)
get_model_covariates(model, strings=TRUE)

## End(Not run)
```

`get_number_of_individuals`
get_number_of_individuals

Description

Retrieve the number of individuals in the model dataset

Usage

```
get_number_of_individuals(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(integer) Number of individuals in the model dataset

Note

For NONMEM models this is the number of individuals of the active dataset, i.e. after filteringof IGNORE and ACCEPT and removal of individuals with no observations.

See Also

get_number_of_observations : Get the number of observations in a dataset

get_number_of_observations_per_individual : Get the number of observations per individual in a dataset

Examples

```
## Not run:
model <- load_example_model("pheno")
get_number_of_individuals(model)

## End(Not run)
```

get_number_of_observations

get_number_of_observations

Description

Retrieve the total number of observations in the model dataset

Usage

```
get_number_of_observations(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(integer) Number of observations in the model dataset

Note

For NONMEM models this is the number of observations of the active dataset, i.e. after filteringof IGNORE and ACCEPT and removal of individuals with no observations.

See Also

get_number_of_individuals : Get the number of individuals in a dataset

get_number_of_observations_per_individual : Get the number of observations per individual in a dataset

Examples

```
## Not run:
model <- load_example_model("pheno")
get_number_of_observations(model)

## End(Not run)
```

get_number_of_observations_per_individual
get_number_of_observations_per_individual

Description

Number of observations for each individual

Usage

```
get_number_of_observations_per_individual(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(data.frame) Number of observations in the model dataset

Note

For NONMEM models this is the individuals and number of observations of the active dataset, i.e.after filtering of IGNORE and ACCEPT and removal of individuals with no observations.

See Also

get_number_of_individuals : Get the number of individuals in a dataset

get_number_of_observations_per_individual : Get the number of observations per individual in a dataset

Examples

```
## Not run:
model <- load_example_model("pheno")
get_number_of_observations_per_individual(model)

## End(Not run)
```

`get_observations` *get_observations*

Description

Get observations from dataset

Usage

```
get_observations(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(data.frame) Observations indexed over ID and TIME

See Also

get_number_of_observations : get the number of observations

get_number_of_observations_per_individual : get the number of observations per individual

Examples

```
## Not run:
model <- load_example_model("pheno")
get_observations(model)

## End(Not run)
```

`get_observation_expression` *get_observation_expression*

Description

Get the full symbolic expression for the observation according to the model

This function currently only support models without ODE systems

Usage

```
get_observation_expression(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Expression) Symbolic expression

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
expr <- get_observation_expression(model)
sympy$pprint(expr)

## End(Not run)
```

---

get_omegas *get_omegas*

Description

Get all omegas (variability parameters) of a model

Usage

```
get_omegas(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Parameters) A copy of all omega parameters

See Also

get_thetas : Get theta parameters

get_sigmas : Get sigma parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
get_omegas(model)

## End(Not run)
```

`get_pk_parameters` *get_pk_parameters*

Description

Retrieves PK parameters in :class:`pharmpy.model`.

Usage

```
get_pk_parameters(model, kind = "all")
```

Arguments

`model`	(Model) Pharmpy model to retrieve the PK parameters from
`kind`	(str) The type of parameter to retrieve: 'absorption', 'distribution', 'elimination', or 'all' (default).

Value

(Parameters) The PK parameters of the given model

See Also

get_individual_parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
sorted(get_pk_parameters(model))
get_pk_parameters(model, 'absorption')
get_pk_parameters(model, 'distribution')
get_pk_parameters(model, 'elimination')

## End(Not run)
```

`get_population_prediction_expression` *get_population_prediction_expression*

Description

Get the full symbolic expression for the modelled population prediction

This function currently only support models without ODE systems

Usage

```
get_population_prediction_expression(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Expression) Symbolic expression

See Also

get_individual_prediction_expression : Get full symbolic epression for the individual prediction

Examples

```
## Not run:
model <- load_example_model("pheno_linear")
get_population_prediction_expression(model)

## End(Not run)
```

`get_sigmas` *get_sigmas*

Description

Get all sigmas (residual error variability parameters) of a model

Usage

```
get_sigmas(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Parameters) A copy of all sigma parameters

See Also

get_thetas : Get theta parameters

get_omegas : Get omega parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
get_sigmas(model)

## End(Not run)
```

---

get_thetas *get_thetas*

Description

Get all thetas (structural parameters) of a model

Usage

```
get_thetas(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Parameters) A copy of all theta parameters

See Also

get_omegas : Get omega parameters

get_sigmas : Get sigma parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
get_thetas(model)

## End(Not run)
```

get_unit_of *get_unit_of*

Description

Derive the physical unit of a variable in the model

Unit information for the dataset needs to be available. The variable can be defined in the code, a dataset olumn, a parameter or a random variable.

Usage

```
get_unit_of(model, variable)
```

Arguments

`model`	(Model) Pharmpy model object
`variable`	(str or Symbol) Find physical unit of this variable

Value

(unit expression) A sympy physics.units expression

Examples

```
## Not run:
model <- load_example_model("pheno")
get_unit_of(model, "Y")
get_unit_of(model, "V")
get_unit_of(model, "WGT")

## End(Not run)
```

greekify_model *greekify_model*

Description

Convert to using greek letters for all population parameters

Usage

```
greekify_model(model, named_subscripts = FALSE)
```

Arguments

`model`	(Model) Pharmpy model
`named_subscripts`	(logical) Use previous parameter names as subscripts. Default is to use integer subscripts

Value

(Model) Reference to the same model

Examples

```
## Not run:
model <- load_example_model("pheno")
model$statements
greekify_model(cleanup_model(model))
model$statements

## End(Not run)
```

`has_additive_error_model`

has_additive_error_model

Description

Check if a model has an additive error model

Usage

```
has_additive_error_model(model)
```

Arguments

`model`	(Model) The model to check

Value

(logical) TRUE if the model has an additive error model and FALSE otherwise

See Also

has_proportional_error_model : Check if a model has a proportional error model

has_combined_error_model : Check if a model has a combined error model

Examples

```
## Not run:
model <- load_example_model("pheno")
has_additive_error_model(model)

## End(Not run)
```

`has_combined_error_model`
has_combined_error_model

Description

Check if a model has a combined additive and proportinal error model

Usage

```
has_combined_error_model(model)
```

Arguments

`model`	(Model) The model to check

Value

(logical) TRUE if the model has a combined error model and FALSE otherwise

See Also

has_additive_error_model : Check if a model has an additive error model

has_proportional_error_model : Check if a model has a proportional error model

Examples

```
## Not run:
model <- load_example_model("pheno")
has_combined_error_model(model)

## End(Not run)
```

`has_first_order_elimination` *has_first_order_elimination*

Description

Check if the model describes first order elimination

This function relies on heuristics and will not be able to detect all possible ways of coding the first order elimination.

Usage

```
has_first_order_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(logical) TRUE if model has describes first order elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
has_first_order_elimination(model)

## End(Not run)
```

`has_michaelis_menten_elimination` *has_michaelis_menten_elimination*

Description

Check if the model describes Michaelis-Menten elimination

This function relies on heuristics and will not be able to detect all possible ways of coding the Michalis-Menten elimination.

Usage

```
has_michaelis_menten_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(logical) TRUE if model has describes Michaelis-Menten elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
has_michaelis_menten_elimination(model)
set_michaelis_menten_elimination(model)
has_michaelis_menten_elimination(model)

## End(Not run)
```

`has_mixed_mm_fo_elimination` *has_mixed_mm_fo_elimination*

Description

Check if the model describes mixed Michaelis-Menten and first order elimination

This function relies on heuristics and will not be able to detect all possible ways of coding the mixed Michalis-Menten and first order elimination.

Usage

```
has_mixed_mm_fo_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(logical) TRUE if model has describes Michaelis-Menten elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
has_mixed_mm_fo_elimination(model)
set_mixed_mm_fo_elimination(model)
has_mixed_mm_fo_elimination(model)

## End(Not run)
```

`has_proportional_error_model` *has_proportional_error_model*

Description

Check if a model has a proportional error model

Usage

```
has_proportional_error_model(model)
```

Arguments

`model`	(Model) The model to check

Value

(logical) TRUE if the model has a proportional error model and FALSE otherwise

See Also

has_additive_error_model : Check if a model has an additive error model

has_combined_error_model : Check if a model has a combined error model

Examples

```
## Not run:
model <- load_example_model("pheno")
has_proportional_error_model(model)

## End(Not run)
```

`has_random_effect` *has_random_effect*

Description

Decides whether the given parameter of a :class:`pharmpy.model` has a random effect.

Usage

```
has_random_effect(model, parameter, level = "all")
```

Arguments

`model`	(Model) Input Pharmpy model
`parameter`	(str) Input parameter
`level`	(str) The variability level to look for: 'iiv', 'iov', or 'all' (default)

Value

(logical) Whether the given parameter has a random effect

See Also

get_individual_parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
has_random_effect(model, 'S1')
has_random_effect(model, 'CL', 'iiv')
has_random_effect(model, 'CL', 'iov')

## End(Not run)
```

---

has_zero_order_absorption

has_zero_order_absorption

Description

Check if ode system describes a zero order absorption

currently defined as having Infusion dose with rate not in dataset

Usage

```
has_zero_order_absorption(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to same model

Examples

```
## Not run:
model <- load_example_model("pheno")
has_zero_order_absorption(model)

## End(Not run)
```

`has_zero_order_elimination`

has_zero_order_elimination

Description

Check if the model describes zero-order elimination

This function relies on heuristics and will not be able to detect all possible ways of coding the zero-order elimination.

Usage

```
has_zero_order_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(logical) TRUE if model has describes zero order elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
has_zero_order_elimination(model)
set_zero_order_elimination(model)
has_zero_order_elimination(model)

## End(Not run)
```

install_pharmpy *Install Pharmpy*

Description

Install the pharmpy-core python package into virtual environment. Uses the same Pharmpy version as pharmr.

Usage

```
install_pharmpy(envname = "r-reticulate", method = "auto")
```

Arguments

envname	(str) name of environment. Default is r-reticulate
method	(str) type of environment type (virtualenv, conda). Default is auto (virtualenv is not available on Windows)

install_pharmpy_devel *Install Pharmpy (with specified version)*

Description

Install the pharmpy-core python package into virtual environment.

Usage

```
install_pharmpy_devel(
  envname = "r-reticulate",
  method = "auto",
  version = "same"
)
```

Arguments

envname	(str) name of environment. Default is r-reticulate
method	(str) type of environment type (virtualenv, conda). Default is auto (virtualenv is not available on Windows)
version	(str) which pharmpy version to use (use 'same' for most cases)

`list_time_varying_covariates` *list_time_varying_covariates*

Description

Return a vector of names of all time varying covariates

Usage

```
list_time_varying_covariates(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(vector) Names of all time varying covariates

See Also

get_covariate_baselines : get baselines for all covariates

Examples

```
## Not run:
model <- load_example_model("pheno")
list_time_varying_covariates(model)

## End(Not run)
```

`load_example_model` *load_example_model*

Description

Load an example model

Load an example model from models built into Pharmpy

Usage

```
load_example_model(name)
```

Arguments

`name`	(str) Name of the model. Currently available models are "pheno" and "pheno_linear"

Value

(Model) Loaded model object

Examples

```
## Not run:
model <- load_example_model("pheno")
model$statements

## End(Not run)
```

make_declarative *make_declarative*

Description

Make the model statments declarative

Each symbol will only be declared once.

Usage

```
make_declarative(model)
```

Arguments

model	(Model) Pharmpy model

Value

(Model) Reference to the same model

Examples

```
## Not run:
model <- load_example_model("pheno")
model$statements$before_odes
make_declarative(model)
model$statements$before_odes

## End(Not run)
```

`mu_reference_model` *mu_reference_model*

Description

Convert model to use mu-referencing

Mu-referencing an eta is to separately define its actual mu (mean) parameter. For example: :math:`CL = \theta_1 e^{\eta_1` with :math:`\eta_1` following a zero-mean normal distribution would give :math:`\mu_1 = log{\theta_1}` and :math:`CL = e^{\mu_1 + \eta_1}`

Usage

```
mu_reference_model(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Model) Reference to same object

Examples

```
## Not run:
model <- load_example_model("pheno")
mu_reference_model(model).statements$before_odes

## End(Not run)
```

`omit_data` *omit_data*

Description

Iterate over omissions of a certain group in a dataset. One group is omitted at a time.

Usage

```
omit_data(dataset_or_model, group, name_pattern = "omitted_{}")
```

Arguments

`dataset_or_model`	(data.frame or Model) Dataset or model for which to omit records
`group`	(str) Name of the column to use for grouping
`name_pattern`	(str) Name to use for generated datasets. A number starting from 1 will be put in the placeholder.

Value

(iterator) Iterator yielding tuples of models/dataframes and the omited group

---

`plot_individual_predictions`
plot_individual_predictions

Description

Plot DV and predictions grouped on individuals

Usage

```
plot_individual_predictions(model, predictions = NULL, individuals = NULL)
```

Arguments

`model`	(Model) Previously run Pharmpy model.
`predictions`	(vector) A vector of names of predictions to plot. NULL for all available
`individuals`	(vector) A vector of individuals to include. NULL for all individuals

Value

(alt.Chart) Plot

`plot_iofv_vs_iofv` *plot_iofv_vs_iofv*

Description

Plot individual OFV of two models against each other

Usage

```
plot_iofv_vs_iofv(model, other)
```

Arguments

`model`	(Model) The first model
`other`	(Model) The second model

Value

(alt.Chart) Scatterplot

`predict_influential_individuals` *predict_influential_individuals*

Description

Predict influential individuals for a model using a machine learning model.

Usage

```
predict_influential_individuals(model, cutoff = 3.84)
```

Arguments

`model`	(Model) Pharmpy model
`cutoff`	(numeric) Cutoff threshold for a dofv signalling an influential individual

Value

(pd.Dataframe) Dataframe over the individuals with a `dofv` column containing the raw predicted delta-OFV and an `influential` column with a boolean to tell whether the individual is influential or not.

See Also

predict_influential_outliers

predict_outliers

`predict_influential_outliers` *predict_influential_outliers*

Description

Predict influential outliers for a model using a machine learning model.

Usage

```
predict_influential_outliers(
  model,
  outlier_cutoff = 3,
  influential_cutoff = 3.84
)
```

Arguments

`model`	(Model) Pharmpy model
`outlier_cutoff`	(numeric) Cutoff threshold for a residual singalling an outlier
`influential_cutoff`	(numeric) Cutoff threshold for a dofv signalling an influential individual

Value

(pd.Dataframe) Dataframe over the individuals with a `outliers` and `dofv` columns containing the raw predictions and `influential`, `outlier` and `influential_outlier` boolean columns.

See Also

predict_influential_individuals

predict_outliers

`predict_outliers` *predict_outliers*

Description

Predict outliers for a model using a machine learning model.

See the :ref:`simeval <Individual OFV summary>` documentation for a definition of the `residual`

Usage

```
predict_outliers(model, cutoff = 3)
```

Arguments

`model`	(Model) Pharmpy model
`cutoff`	(numeric) Cutoff threshold for a residual singalling an outlier

Value

(pd.Dataframe) Dataframe over the individuals with a `residual` column containing the raw predicted residuals and a `outlier` column with a boolean to tell whether the individual is an outlier or not.

See Also

predict_influential_individuals

predict_influential_outliers

Examples

```
## Not run:
model <- load_example_model("pheno")
predict_outliers(model)

## End(Not run)
```

`print_fit_summary` *print_fit_summary*

Description

Print a summary of the model fit

Usage

```
print_fit_summary(model)
```

Arguments

`model`	(Model) Pharmpy model object

print_model_code *print_model_code*

Description

Print the model code of the underlying model language

Usage

```
print_model_code(model)
```

Arguments

model	(Model) Pharmpy model

Examples

```
## Not run:
model <- load_example_model("pheno")
print_model_code(model)

## End(Not run)
```

print_model_symbols *print_model_symbols*

Description

Print all symbols defined in a model

Symbols will be in one of the categories thetas, etas, omegas, epsilons, sigmas, variables and data columns

Usage

```
print_model_symbols(model)
```

Arguments

model	(Model) Pharmpy model object

Examples

```
## Not run:
model <- load_example_model("pheno")
print_model_symbols(model)

## End(Not run)
```

`print_pharmpy_version` *Print pharmpy version*

Description

Print the pharmpy version pharmr uses.

Usage

```
print_pharmpy_version()
```

`rank_models` *rank_models*

Description

Ranks a vector of models

Ranks a vector of models with a given ranking function

Usage

```
rank_models(
  base_model,
  models,
  strictness = NULL,
  rank_type = "ofv",
  cutoff = NULL,
  bic_type = "mixed"
)
```

Arguments

`base_model`	(Model) Base model to compare to
`models`	(vector) List of models
`strictness`	(vector or NULL) List of strictness criteria to be fulfilled, currently only minimization successful. Default is NULL
`rank_type`	(str) Name of ranking type. Available options are 'ofv', 'aic', 'bic', 'lrt' (OFV with LRT)
`cutoff`	(numeric or NULL) Value to use as cutoff. If using LRT, cutoff denotes p-value. Default is NULL
`bic_type`	(str) Type of BIC to calculate. Default is the mixed effects.

Value

((data.frame, vector)) A tuple with a DataFrame of the ranked models and a vector of ranked models sorted by rank

Examples

```
## Not run:
model_1 <- load_example_model("pheno")
model_2 <- load_example_model("pheno_linear")
rank_models(model_1, c(model_2),
            strictness=c('minimization_successful'),
            rank_type='lrt')

## End(Not run)
```

`read_dataset_from_datainfo`
read_dataset_from_datainfo

Description

Read a dataset given a datainfo object or path to a datainfo file

Usage

```
read_dataset_from_datainfo(datainfo)
```

Arguments

`datainfo`	(DataInfo | Path | str) A datainfo object or a path to a datainfo object

Value

(data.frame) The dataset

read_model *read_model*

Description

Read model from file

Usage

```
read_model(path)
```

Arguments

path	(str or Path) Path to model

Value

(Model) Read model object

See Also

read_model_from_database : Read model from database

read_model_from_string : Read model from string

Examples

```
## Not run:
model <- read_model("/home/run1$mod")

## End(Not run)
```

read_model_from_database *read_model_from_database*

Description

Read model from model database

Usage

```
read_model_from_database(name, database = NULL)
```

Arguments

name	(str) Name of model to use as lookup
database	(Database) Database to use. Will use default database if not specified.

Value

(Model) Read model object

See Also

read_model : Read model from file

read_model_from_string : Read model from string

Examples

```
## Not run:
model <- read_model_from_database("run1")

## End(Not run)
```

`read_model_from_string` *read_model_from_string*

Description

Read model from the model code in a string

Usage

```
read_model_from_string(code, path = NULL)
```

Arguments

`code`	(str) Model code to read
`path`	(Path or str) Specified to set the path for the created model

Value

(Model) Pharmpy model object

See Also

read_model : Read model from file

read_model_from_database : Read model from database

Examples

```
## Not run:
s <- "$PROBLEM
$INPUT ID DV TIME
$DATA file$csv
$PRED
Y=THETA(1)+ETA(1)+ERR(1)
$THETA 1
$OMEGA 0.1
$SIGMA 1
$ESTIMATION METHOD=1"
read_model_from_string(s)

## End(Not run)
```

`read_results` *read_results*

Description

Read results object from file

Usage

```
read_results(path)
```

Arguments

`path`	(str, Path) Path to results file

Value

(Results) Results object for tool

See Also

create_results

Examples

```
## Not run:
res <- read_resuts("results$json")

## End(Not run)
```

remove_covariance_step

remove_covariance_step

Description

Removes covariance step to the final estimation step

Usage

```
remove_covariance_step(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to the same model object

See Also

add_estimation_step

set_estimation_step

remove_estimation_step

append_estimation_step_options

add_covariance_step

set_evaluation_step

Examples

```
## Not run:
model <- load_example_model("pheno")
remove_covariance_step(model)
ests <- model$estimation_steps
ests[1]

## End(Not run)
```

`remove_error_model` *remove_error_model*

Description

Remove error model.

Usage

```
remove_error_model(model)
```

Arguments

`model`	(Model) Remove error model for this model

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
model$statements$find_assignment("Y")
remove_error_model(model)
model$statements$find_assignment("Y")

## End(Not run)
```

`remove_estimation_step` *remove_estimation_step*

Description

Remove estimation step

Usage

```
remove_estimation_step(model, idx)
```

Arguments

`model`	(Model) Pharmpy model
`idx`	(integer) index of estimation step to remove (starting from 0)

Value

(Model) Reference to the same model object

See Also

add_estimation_step

set_estimation_step

append_estimation_step_options

add_covariance_step

remove_covariance_step

set_evaluation_step

Examples

```
## Not run:
model <- load_example_model("pheno")
remove_estimation_step(model, 0)
ests <- model$estimation_steps
length(ests)

## End(Not run)
```

remove_iiv *remove_iiv*

Description

Removes all IIV etas given a vector with eta names and/or parameter names.

Usage

```
remove_iiv(model, to_remove = NULL)
```

Arguments

model	(Model) Pharmpy model to create block effect on.
to_remove	(str, vector) Name/names of etas and/or name/names of individual parameters to remove. If NULL, all etas that are IIVs will be removed. NULL is default.

Value

(Model) Reference to the same model

See Also

remove_iov

add_iiv

add_iov

add_pk_iiv

Examples

```
## Not run:
model <- load_example_model("pheno")
remove_iiv(model)
model$statements$find_assignment("CL")
model <- load_example_model("pheno")
remove_iiv(model, "V")
model$statements$find_assignment("V")

## End(Not run)
```

remove_iov *remove_iov*

Description

Removes all IOV etas given a vector with eta names.

Usage

```
remove_iov(model, to_remove = NULL)
```

Arguments

`model`	(Model) Pharmpy model to remove IOV from.
`to_remove`	(str, vector) Name/names of IOV etas to remove, e.g. 'ETA_IOV_11'. If NULL, all etas that are IOVs will be removed. NULL is default.

Value

(Model) Reference to the same model

See Also

add_iiv

add_iov

remove_iiv

add_pk_iiv

Examples

```
## Not run:
model <- load_example_model("pheno")
remove_iov(model)

## End(Not run)
```

---

remove_lag_time *remove_lag_time*

Description

Remove lag time from the dose compartment of model.

Usage

```
remove_lag_time(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to same model

See Also

set_transit_compartments

add_lag_time

Examples

```
## Not run:
model <- load_example_model("pheno")
remove_lag_time(model)

## End(Not run)
```

`remove_loq_data` *remove_loq_data*

Description

Remove loq data records from the dataset

Does nothing if none of the limits is specified.

Usage

```
remove_loq_data(model, lloq = NULL, uloq = NULL)
```

Arguments

`model`	(Model) Pharmpy model object
`lloq`	(numeric) Lower limit of quantification. Default not specified.
`uloq`	(numeric) Upper limit of quantification. Default not specified.

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
remove_loq_data(model, lloq=10, uloq=40)
length(model$dataset)

## End(Not run)
```

`remove_peripheral_compartment` *remove_peripheral_compartment*

Description

Remove a peripheral distribution compartment from model

Initial estimates:

== == n == ==================================
2 :math:`{CL} = {CL'}`, :math:`{QP1} = {CL'}` and :math:`{VP1} = {VC'} * 0.05` 3 :math:`{QP1} = ({QP1'} + {QP2'}) / 2`, :math:`{VP1} = {VP1'} + {VP2'}` == ==

Usage

```
remove_peripheral_compartment(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to same model

See Also

set_peripheral_compartment

add_peripheral_compartment

Examples

```
## Not run:
model <- load_example_model("pheno")
set_peripheral_compartments(model, 2)
remove_peripheral_compartment(model)
model$statements$ode_system

## End(Not run)
```

`remove_unused_parameters_and_rvs` *remove_unused_parameters_and_rvs*

Description

Remove any parameters and rvs that are not used in the model statements

Usage

```
remove_unused_parameters_and_rvs(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Model) Reference to same model object

`rename_symbols` *rename_symbols*

Description

Rename symbols in the model

Make sure that no name clash occur.

Usage

```
rename_symbols(model, new_names)
```

Arguments

`model`	(Model) Pharmpy model object
`new_names`	(list) From old name or symbol to new name or symbol

Value

(Model) Reference to same model object

`resample_data` *resample_data*

Description

Iterate over resamples of a dataset.

The dataset will be grouped on the group column then groups will be selected randomly with or without replacement to form a new dataset. The groups will be renumbered from 1 and upwards to keep them separated in the new dataset.

Usage

```
resample_data(
  dataset_or_model,
  group,
  resamples = 1,
  stratify = NULL,
  sample_size = NULL,
  replace = FALSE,
  name_pattern = "resample_{}",
  name = NULL
)
```

Arguments

`dataset_or_model`	(data.frame or Model) Dataset or Model to use
`group`	(str) Name of column to group by
`resamples`	(integer) Number of resamples (iterations) to make
`stratify`	(str) Name of column to use for stratification. The values in the stratification column must be equal within a group so that the group can be uniquely determined. A ValueError exception will be raised otherwise.
`sample_size`	(integer) The number of groups that should be sampled. The default is the number of groups. If using stratification the default is to sample using the proportion of the stratas in the dataset. A list of specific sample sizes for each strata can also be supplied.
`replace`	(logical) A boolean controlling whether sampling should be done with or without replacement
`name_pattern`	(str) Name to use for generated datasets. A number starting from 1 will be put in the placeholder.
`name`	(str) Option to name pattern in case of only one resample

Value

(iterator) An iterator yielding tuples of a resampled DataFrame and a vector of resampled groups in order

---

`reset_index` *Reset index*

Description

Reset index of dataframe.

Reset index from a multi indexed data.frame so that index is added as columns

Usage

```
reset_index(df)
```

Arguments

`df`	A data.frame converted from python using reticulate

`retrieve_models` *retrieve_models*

Description

Retrieve models after a tool runs

Any models created and run by the tool can be retrieved.

Usage

```
retrieve_models(path, names = NULL)
```

Arguments

`path`	(str or Path) A path to the tool directory
`names`	(vector) List of names of the models to retrieve or NULL for all

Value

(vector) List of retrieved model objects

`run_allometry` *run_allometry*

Description

Run allometry tool. For more details, see :ref:`allometry`.

Usage

```
run_allometry(
  model = NULL,
  allometric_variable = "WT",
  reference_value = 70,
  parameters = NULL,
  initials = NULL,
  lower_bounds = NULL,
  upper_bounds = NULL,
  fixed = TRUE
)
```

Arguments

`model`	(Model) Pharmpy model
`allometric_variable`	(str) Name of the variable to use for allometric scaling (default is WT)
`reference_value`	(numeric) Reference value for the allometric variable (default is 70)
`parameters`	(vector) Parameters to apply scaling to (default is all CL, Q and V parameters)
`initials`	(vector) Initial estimates for the exponents. (default is to use 0.75 for CL and Qs and 1 for Vs)
`lower_bounds`	(vector) Lower bounds for the exponents. (default is 0 for all parameters)
`upper_bounds`	(vector) Upper bounds for the exponents. (default is 2 for all parameters)
`fixed`	(logical) Should the exponents be fixed or not. (default TRUE)

Value

(AllometryResults) Allometry tool result object

Examples

```
## Not run:
model <- load_example_model("pheno")
run_allometry(model=model, allometric_variable='WGT')

## End(Not run)
```

---

run_amd *run_amd*

Description

Run Automatic Model Development (AMD) tool

Runs structural modelsearch, IIV building, and resmod

Usage

```
run_amd(
  input,
  modeltype = "pk_oral",
  cl_init = 0.01,
  vc_init = 1,
  mat_init = 0.1,
  search_space = NULL,
  lloq = NULL,
  order = NULL,
  categorical = NULL,
  continuous = NULL,
  allometric_variable = NULL,
  occasion = NULL
)
```

Arguments

`input`	(Model) Read model object/Path to a dataset
`modeltype`	(str) Type of model to build. Either 'pk_oral' or 'pk_iv'
`cl_init`	(numeric) Initial estimate for the population clearance
`vc_init`	(numeric) Initial estimate for the central compartment population volume
`mat_init`	(numeric) Initial estimate for the mean absorption time (not for iv models)
`search_space`	(str) MFL for search space for structural model
`lloq`	(numeric) Lower limit of quantification. LOQ data will be removed.
`order`	(vector) Runorder of components
`categorical`	(vector) List of categorical covariates
`continuous`	(vector) List of continuous covariates
`allometric_variable`	(str or Symbol) Variable to use for allometry
`occasion`	(str) Name of occasion column

Value

(Model) Reference to the same model object

See Also

run_iiv

run_tool

Examples

```
## Not run:
model <- load_example_model("pheno")
run_amd(model)

## End(Not run)
```

run_covsearch *run_covsearch*

Description

Run COVsearch tool. For more details, see :ref:`covsearch`.

Usage

```
run_covsearch(
  effects,
  p_forward = 0.05,
  max_steps = -1,
  algorithm = "scm-forward",
  model = NULL
)
```

Arguments

`effects`	(str | vector) The vector of candidates to try, either in DSL str form or in (optionally compact) tuple form.
`p_forward`	(numeric) The p-value to use in the likelihood ratio test for forward steps
`max_steps`	(integer) The maximum number of search steps to make
`algorithm`	(str) The search algorithm to use. Currently only 'scm-forward' is supported.
`model`	(Model) Pharmpy model

Value

(COVSearchResults) COVsearch tool result object

Examples

```
## Not run:
model <- load_example_model("pheno")
res <- run_covsearch([

## End(Not run)
```

run_iivsearch *run_iivsearch*

Description

Run IIVsearch tool. For more details, see :ref:`iivsearch`.

Usage

```
run_iivsearch(
  algorithm,
  iiv_strategy = "no_add",
  rank_type = "bic",
  cutoff = NULL,
  model = NULL
)
```

Arguments

`algorithm`	(str) Which algorithm to run (brute_force, brute_force_no_of_etas, brute_force_block_structure)
`iiv_strategy`	(str) If/how IIV should be added to start model. Possible strategies are 'no_add', 'add_diagonal', or 'fullblock'. Default is 'no_add'
`rank_type`	(str) Which ranking type should be used (OFV, AIC, BIC). Default is BIC
`cutoff`	(numeric) Cutoff for which value of the ranking function that is considered significant. Default is NULL (all models will be ranked)
`model`	(Model) Pharmpy model

Value

(IIVSearchResults) IIVsearch tool result object

Examples

```
## Not run:
model <- load_example_model("pheno")
run_iivsearch('brute_force', model=model)

## End(Not run)
```

`run_iovsearch` *run_iovsearch*

Description

Run IOVsearch tool. For more details, see :ref:`iovsearch`.

Usage

```
run_iovsearch(
  column = "OCC",
  list_of_parameters = NULL,
  rank_type = "bic",
  cutoff = NULL,
  distribution = "same-as-iiv",
  model = NULL
)
```

Arguments

`column`	(str) Name of column in dataset to use as occasion column (default is 'OCC')
`list_of_parameters`	(vector) List of parameters to test IOV on, if none all parameters with IIV will be tested (default)
`rank_type`	(str) Which ranking type should be used (OFV, AIC, BIC). Default is BIC
`cutoff`	(NULL or numeric) Cutoff for which value of the ranking type that is considered significant. Default is NULL (all models will be ranked)
`distribution`	(str) Which distribution added IOVs should have (default is same-as-iiv)
`model`	(Model) Pharmpy model

Value

(IOVSearchResults) IOVSearch tool result object

Examples

```
## Not run:
model <- load_example_model("pheno")
run_iovsearch('OCC', model=model)

## End(Not run)
```

run_modelsearch *run_modelsearch*

Description

Run Modelsearch tool. For more details, see :ref:`modelsearch`.

Usage

```
run_modelsearch(
  search_space,
  algorithm,
  iiv_strategy = "absorption_delay",
  rank_type = "bic",
  cutoff = NULL,
  model = NULL
)
```

Arguments

`search_space`	(str) Search space to test
`algorithm`	(str) Algorithm to use (e.g. exhaustive)
`iiv_strategy`	(str) If/how IIV should be added to candidate models. Possible strategies are 'no_add', 'add_diagonal', 'fullblock', or 'absorption_delay'. Default is 'absorption_delay'
`rank_type`	(str) Which ranking type should be used (OFV, AIC, BIC). Default is BIC
`cutoff`	(numeric) Cutoff for which value of the ranking function that is considered significant. Default is NULL (all models will be ranked)
`model`	(Model) Pharmpy model

Value

(ModelSearchResults) Modelsearch tool result object

Examples

```
## Not run:
model <- load_example_model("pheno")
run_modelsearch('ABSORPTION(ZO);PERIPHERALS(1)', 'exhaustive', model=model)

## End(Not run)
```

run_resmod *run_resmod*

Description

Run the resmod tool. For more details, see :ref:`resmod`.

Usage

```
run_resmod(model = NULL, groups = 4, p_value = 0.05, skip = NULL)
```

Arguments

`model`	(Model) Pharmpy model
`groups`	(integer) The number of bins to use for the time varying models
`p_value`	(numeric) The p-value to use for the likelihood ratio test
`skip`	(vector) A vector of models to not attempt.

Value

(ResmodResults) Resmod tool result object

Examples

```
## Not run:
model <- load_example_model("pheno")
run_resmod(model=model)

## End(Not run)
```

run_tool *run_tool*

Description

Run tool workflow

Usage

```
run_tool(name, ...)
```

Arguments

`name`	(str) Name of tool to run
`...`	Arguments to pass to tool

Value

(Results) Results object for tool

Examples

```
## Not run:
model <- load_example_model("pheno")
res <- run_tool("resmod", model)

## End(Not run)
```

---

sample_individual_estimates
sample_individual_estimates

Description

Sample individual estimates given their covariance.

Usage

```
sample_individual_estimates(
  model,
  parameters = NULL,
  samples_per_id = 100,
  rng = NULL
)
```

Arguments

`model`	(Model) Pharmpy model
`parameters`	(vector) A vector of a subset of individual parameters to sample. Default is NULL, which means all.
`samples_per_id`	(integer) Number of samples per individual
`rng`	(rng or integer) Random number generator or seed

Value

(data.frame) Pool of samples in a DataFrame

See Also

sample_parameters_from_covariance_matrix : Sample parameter vectors using the

uncertainty covariance matrix

sample_parameters_uniformly : Sample parameter vectors using uniform distribution

Examples

```
## Not run:
model <- load_example_model("pheno")
rng <- create_rng(23)
sample_individual_estimates(model, samples_per_id=2, rng=rng)

## End(Not run)
```

---

sample_parameters_from_covariance_matrix

sample_parameters_from_covariance_matrix

Description

Sample parameter vectors using the covariance matrix

If modelfit_results is not provided the results from the model will be used

Usage

```
sample_parameters_from_covariance_matrix(
  model,
  modelfit_results = NULL,
  parameters = NULL,
  force_posdef_samples = NULL,
  force_posdef_covmatrix = FALSE,
  n = 1,
  rng = NULL
)
```

Arguments

`model`	(Model) Input model
`modelfit_results`	(ModelfitResults) Alternative results object. Default is to use the one in model
`parameters`	(vector) Use to only sample a subset of the parameters. NULL means all
`force_posdef_samples`	(integer) Set to how many iterations to do before forcing all samples to be positive definite. NULL is default and means never and 0 means always
`force_posdef_covmatrix`	(logical) Set to TRUE to force the input covariance matrix to be positive definite
`n`	(integer) Number of samples
`rng`	(Generator) Random number generator

Value

(data.frame) A dataframe with one sample per row

See Also

sample_parameters_uniformly : Sample parameter vectors using uniform distribution

sample_individual_estimates : Sample individual estiates given their covariance

Examples

```
## Not run:
model <- load_example_model("pheno")
rng <- create_rng(23)
sample_parameters_from_covariance_matrix(model, n=3, rng=rng)

## End(Not run)
```

`sample_parameters_uniformly`
sample_parameters_uniformly

Description

Sample parameter vectors using uniform sampling

Each parameter value will be randomly sampled from a uniform distribution with the bounds being estimate ± estimate * fraction.

Usage

```
sample_parameters_uniformly(
  model,
  fraction = 0.1,
  parameters = NULL,
  force_posdef_samples = NULL,
  n = 1,
  rng = NULL
)
```

Arguments

`model`	(Model) Pharmpy model
`fraction`	(numeric) Fraction of estimate value to use for distribution bounds
`parameters`	(data.frame) Names of parameters to use. Default is to use all parameters in the model.
`force_posdef_samples`	(integer) Number of samples to reject before forcing variability parameters to give positive definite covariance matrices.
`n`	(integer) Number of samples
`rng`	(integer or rng) Random number generator or seed

Value

(data.frame) samples

See Also

sample_parameters_from_covariance_matrix : Sample parameter vectors using the

uncertainty covariance matrix

sample_individual_estimates : Sample individual estiates given their covariance

Examples

```
## Not run:
model <- load_example_model("pheno")
rng <- create_rng(23)
sample_parameters_uniformly(model, n=3, rng=rng)

## End(Not run)
```

`set_additive_error_model` *set_additive_error_model*

Description

Set an additive error model. Initial estimate for new sigma is :math:`(min(DV)/2)²`.

The error function being applied depends on the data transformation. The table displays some examples.

+———————+——————————————-+ | Data transformation | Additive error |
+=======================+==+ | :math:y
| :math:f + epsilon_1 | +———————+——————————————-+ | :math:`log(y)` |
:math:`log(f) + frac{epsilon_1}{f}` | +———————+——————————————-+

Usage

```
set_additive_error_model(model, data_trans = NULL, series_terms = 2)
```

Arguments

`model`	(Model) Set error model for this model
`data_trans`	(str or expression) A data transformation expression or NULL (default) to use the transformation specified by the model. Series expansion will be used for approximation.
`series_terms`	(integer) Number of terms to use for the series expansion approximation for data transformation.

Value

(Model) Reference to the same model object

See Also

set_proportional_error_model : Proportional error model

set_combined_error_model : Combined error model

Examples

```
## Not run:
model <- load_example_model("pheno")
model$statements$find_assignment("Y")
set_additive_error_model(model)
model$statements$find_assignment("Y")
model <- load_example_model("pheno")
model$statements$find_assignment("Y")
set_additive_error_model(model, data_trans="log(Y)")
model$statements$find_assignment("Y")

## End(Not run)
```

`set_bolus_absorption` *set_bolus_absorption*

Description

Set or change to bolus absorption rate.

Currently lagtime together with bolus absorption is not supported.

Usage

```
set_bolus_absorption(model)
```

Arguments

`model`	(Model) Model to set or change absorption rate

Value

(Model) Reference to same model

See Also

set_zero_order_absorption

set_first_order_absorption

Examples

```
## Not run:
model <- load_example_model("pheno")
set_bolus_absorption(model)
model$statements$ode_system

## End(Not run)
```

`set_combined_error_model` *set_combined_error_model*

Description

Set a combined error model. Initial estimates for new sigmas are :math:`(min(DV)/2)²` for proportional and 0.09 for additive.

The error function being applied depends on the data transformation.

+———————————+————————————————————————+ | Data transformation | Combined error | +=======================+===

| :math:y | :math:`f + f epsilon_1 + epsilon_2` | +————————————+———————————————————————————————+ | :math:`log(y)` | :math:`log(f) + epsilon_1 + frac{epsilon_2}{f}` | +———————————+————————————————————————+

Usage

```
set_combined_error_model(model, data_trans = NULL)
```

Arguments

`model`	(Model) Set error model for this model
`data_trans`	(str or expression) A data transformation expression or NULL (default) to use the transformation specified by the model.

Value

(Model) Reference to the same model

See Also

set_additive_error_model : Additive error model

set_proportional_error_model: Proportional error model

Examples

```
## Not run:
model <- remove_error_model(load_example_model("pheno"))
set_combined_error_model(model)
model$statements$find_assignment("Y")
model <- remove_error_model(load_example_model("pheno"))
set_combined_error_model(model, data_trans="log(Y)")
model$statements$find_assignment("Y")

## End(Not run)
```

`set_dtbs_error_model` *set_dtbs_error_model*

Description

Dynamic transform both sides

Usage

```
set_dtbs_error_model(model, fix_to_log = FALSE)
```

Arguments

`model`	(Model) Pharmpy model
`fix_to_log`	(Boolean) Set to TRUE to fix lambda and zeta to 0, i.e. emulating log-transformed data

Value

(Model) Reference to the same model

Examples

```
## Not run:
model <- load_example_model("pheno")
set_dtbs_error_model(model)

## End(Not run)
```

set_estimation_step *set_estimation_step*

Description

Set estimation step

Sets estimation step for a model. Methods currently supported are: FO, FOCE, ITS, LAPLACE, IMPMAP, IMP, SAEM, BAYES

Usage

```
set_estimation_step(model, method, idx = 0, ...)
```

Arguments

`model`	(Model) Pharmpy model
`method`	(str) estimation method to change to
`idx`	(integer) index of estimation step, default is 0 (first estimation step)
`...`	Arguments to pass to EstimationStep (such as interaction, evaluation)

Value

(Model) Reference to the same model object

See Also

add_estimation_step

remove_estimation_step

append_estimation_step_options

add_covariance_step

remove_covariance_step

set_evaluation_step

Examples

```
## Not run:
model <- load_example_model("pheno")
opts <- list('NITER'=1000, 'ISAMPLE'=100)
set_estimation_step(model, "IMP", evaluation=TRUE, tool_options=opts)
model$estimation_steps[1]

## End(Not run)
```

`set_evaluation_step` *set_evaluation_step*

Description

Set estimation step

Sets estimation step for a model. Methods currently supported are: FO, FOCE, ITS, LAPLACE, IMPMAP, IMP, SAEM, BAYES

Usage

```
set_evaluation_step(model, idx = -1)
```

Arguments

`model`	(Model) Pharmpy model
`idx`	(integer) index of estimation step, default is -1 (last estimation step)

Value

(Model) Reference to the same model object

See Also

set_estimation_step

add_estimation_step

remove_estimation_step

append_estimation_step_options

add_covariance_step

remove_covariance_step

Examples

```
## Not run:
model <- load_example_model("pheno")
set_evaluation_step(model)
model$estimation_steps[1]

## End(Not run)
```

`set_first_order_absorption` *set_first_order_absorption*

Description

Set or change to first order absorption rate.

Initial estimate for absorption rate is set to the previous rate if available, otherwise it is set to the time of first observation/2.

Usage

```
set_first_order_absorption(model)
```

Arguments

`model`	(Model) Model to set or change to use first order absorption rate

Value

(Model) Reference to same model

See Also

set_bolus_order_absorption

set_zero_order_absorption

Examples

```
## Not run:
model <- load_example_model("pheno")
set_first_order_absorption(model)
model$statements$ode_system

## End(Not run)
```

`set_first_order_elimination` *set_first_order_elimination*

Description

Sets elimination to first order

Usage

```
set_first_order_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to same model

See Also

set_zero_order_elimination

set_michaelis_menten_elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
set_first_order_elimination(model)
model$statements$ode_system

## End(Not run)
```

`set_iiv_on_ruv` *set_iiv_on_ruv*

Description

Multiplies epsilons with exponential (new) etas.

Initial variance for new etas is 0.09.

Usage

```
set_iiv_on_ruv(model, list_of_eps = NULL, same_eta = TRUE, eta_names = NULL)
```

Arguments

`model`	(Model) Pharmpy model to apply IIV on epsilons.
`list_of_eps`	(str, vector) Name/names of epsilons to multiply with exponential etas. If NULL, all epsilons will be chosen. NULL is default.
`same_eta`	(logical) Boolean of whether all RUVs from input should use the same new ETA or if one ETA should be created for each RUV. TRUE is default.
`eta_names`	(str, vector) Custom names of new etas. Must be equal to the number epsilons or 1 if same eta.

Value

(Model) Reference to same model

See Also

set_power_on_ruv

Examples

```
## Not run:
model <- load_example_model("pheno")
set_iiv_on_ruv(model)
model$statements$find_assignment("Y")

## End(Not run)
```

---

set_initial_estimates *set_initial_estimates*

Description

Set initial estimates

Usage

```
set_initial_estimates(model, inits)
```

Arguments

`model`	(Model) Pharmpy model
`inits`	(list) A list of parameter init for parameters to change

Value

(Model) Reference to the same model object

See Also

fix_parameters_to : Fixing and setting parameter initial estimates in the same function
unfix_paramaters_to : Unfixing parameters and setting a new initial estimate in the same

Examples

```
## Not run:
model <- load_example_model("pheno")
set_initial_estimates(model, list('THETA(1)'=2))
model$parameters['THETA(1)']

## End(Not run)
```

`set_lower_bounds` *set_lower_bounds*

Description

Set parameter lower bounds

Usage

```
set_lower_bounds(model, bounds)
```

Arguments

`model`	(Model) Pharmpy model
`bounds`	(list) A list of parameter bounds for parameters to change

Value

(Model) Reference to the same model object

See Also

set_upper_bounds : Set parameter upper bounds

unconstrain_parameters : Remove all constraints of parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
set_lower_bounds(model, {'THETA(1)': -10})
model$parameters['THETA(1)']

## End(Not run)
```

`set_michaelis_menten_elimination` *set_michaelis_menten_elimination*

Description

Sets elimination to Michaelis-Menten.

Initial estimate for CLMM is set to CL and KM is set to :math:`2*max(DV)`.

Usage

```
set_michaelis_menten_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to the same model

See Also

set_first_order_elimination

set_zero_order_elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
set_michaelis_menten_elimination(model)
model$statements$ode_system

## End(Not run)
```

`set_mixed_mm_fo_elimination` *set_mixed_mm_fo_elimination*

Description

Sets elimination to mixed Michaelis-Menten and first order.

Initial estimate for CLMM is set to CL/2 and KM is set to :math:`2*max(DV)`.

Usage

```
set_mixed_mm_fo_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to the same model

See Also

set_first_order_elimination

set_zero_order_elimination

set_michaelis_menten_elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
set_mixed_mm_fo_elimination(model)
model$statements$ode_system

## End(Not run)
```

---

set_name *set_name*

Description

Set name of model object

Usage

```
set_name(model, new_name)
```

Arguments

model	(Model) Pharmpy model
new_name	(str) New name of model

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
model$name
set_name(model, "run2")
model$name

## End(Not run)
```

`set_ode_solver` *set_ode_solver*

Description

Sets ODE solver to use for model

Recognized solvers and their corresponding NONMEM advans:

+——————————-+———————+ | Solver | NONMEM ADVAN | +==========================+======
| CVODES | ADVAN14 | +——————————-+———————+ | DGEAR | ADVAN8 | +——
————————-+———————+ | DVERK | ADVAN6 | +——————————-+——————
—+ | IDA | ADVAN15 | +——————————-+———————+ | LSODA | ADVAN13 | +——
————————-+———————+ | LSODI | ADVAN9 | +——————————-+——————
—+

Usage

```
set_ode_solver(model, solver)
```

Arguments

`model`	(Model) Pharmpy model
`solver`	(str) Solver to use or NULL for no preference

Value

(Model) Reference to same model

Examples

```
## Not run:
model <- load_example_model("pheno")
set_ode_solver(model, 'LSODA')

## End(Not run)
```

`set_peripheral_compartments` *set_peripheral_compartments*

Description

Sets the number of peripheral compartments to a specified number.

Usage

```
set_peripheral_compartments(model, n)
```

Arguments

`model`	(Model) Pharmpy model
`n`	(integer) Number of transit compartments

Value

(Model) Reference to same model

See Also

add_peripheral_compartment

remove_peripheral_compartment

Examples

```
## Not run:
model <- load_example_model("pheno")
set_peripheral_compartments(model, 2)
model$statements$ode_system

## End(Not run)
```

`set_power_on_ruv` *set_power_on_ruv*

Description

Applies a power effect to provided epsilons.

Initial estimates for new thetas are 1 if the error model is proportional, otherwise they are 0.1.

Usage

```
set_power_on_ruv(
  model,
  list_of_eps = NULL,
  lower_limit = 0.01,
  ipred = NULL,
  zero_protection = FALSE
)
```

Arguments

`model`	(Model) Pharmpy model to create block effect on.
`list_of_eps`	(str, vector) Name/names of epsilons to apply power effect. If NULL, all epsilons will be used. NULL is default.
`lower_limit`	(integer or NULL) Lower limit of power (theta). NULL for no limit.
`ipred`	(Symbol) Symbol to use as IPRED. Default is to autodetect expression for IPRED.
`zero_protection`	(logical) Set to TRUE to add code protecting from IPRED=0

Value

(Model) Reference to the same model

See Also

set_iiv_on_ruv

Examples

```
## Not run:
model <- load_example_model("pheno")
set_power_on_ruv(model)
model$statements$find_assignment("Y")

## End(Not run)
```

`set_proportional_error_model`
set_proportional_error_model

Description

Set a proportional error model. Initial estimate for new sigma is 0.09.

The error function being applied depends on the data transformation.

+————————+———————————————-+ | Data transformation | Proportional error | +=======================+======================================+ | :math:y | :math:`f + f epsilon_1` | +————————+———————————————-+ | :math:`log(y)` | :math:`log(f) + epsilon_1` | +————————+———————————————-+

Usage

```
set_proportional_error_model(model, data_trans = NULL, zero_protection = FALSE)
```

Arguments

`model`	(Model) Set error model for this model
`data_trans`	(str or expression) A data transformation expression or NULL (default) to use the transformation specified by the model.
`zero_protection`	(logical) Set to TRUE to add code protecting from IPRED=0

Value

(Model) Reference to the same model object

See Also

set_additive_error_model : Additive error model

set_combined_error_model : Combined error model

Examples

```
## Not run:
model <- remove_error_model(load_example_model("pheno"))
set_proportional_error_model(model)
model$statements$find_assignment("Y")
model <- remove_error_model(load_example_model("pheno"))
set_proportional_error_model(model, data_trans="log(Y)", zero_protection=TRUE)
model$statements$after_odes

## End(Not run)
```

`set_seq_zo_fo_absorption` *set_seq_zo_fo_absorption*

Description

Set or change to sequential zero order first order absorption rate.

Initial estimate for absorption rate is set the previous rate if available, otherwise it is set to the time of first observation/2.

Currently lagtime together with sequential zero order first order absorption is not supported.

Usage

```
set_seq_zo_fo_absorption(model)
```

Arguments

`model`	(Model) Model to set or change absorption rate

Value

(Model) Reference to same model

See Also

set_bolus_order_absorption

set_zero_order_absorption

set_first_order_absorption

Examples

```
## Not run:
model <- load_example_model("pheno")
set_seq_zo_fo_absorption(model)
model$statements$ode_system

## End(Not run)
```

---

set_time_varying_error_model

set_time_varying_error_model

Description

Set a time varying error model per time cutoff

Usage

```
set_time_varying_error_model(model, cutoff, idv = "TIME")
```

Arguments

`model`	(Model) Pharmpy model
`cutoff`	(numeric) A value at the given quantile over idv column
`idv`	(str) Time or time after dose, default is Time

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
set_time_varying_error_model(model, cutoff=1.0)
model$statements$find_assignment("Y")

## End(Not run)
```

set_transit_compartments *set_transit_compartments*

Description

Set the number of transit compartments of model.

Initial estimate for absorption rate is set the previous rate if available, otherwise it is set to the time of first observation/2.

Usage

```
set_transit_compartments(model, n, keep_depot = TRUE)
```

Arguments

`model`	(Model) Pharmpy model
`n`	(integer) Number of transit compartments
`keep_depot`	(logical) FALSE to convert depot compartment into a transit compartment

Value

(Model) Reference to same model

See Also

add_lag_time

Examples

```
## Not run:
model <- load_example_model("pheno")
set_transit_compartments(model, 3)
model$statements$ode_system

## End(Not run)
```

set_upper_bounds *set_upper_bounds*

Description

Set parameter upper bounds

Usage

```
set_upper_bounds(model, bounds)
```

Arguments

model	(Model) Pharmpy model
bounds	(list) A list of parameter bounds for parameters to change

Value

(Model) Reference to the same model object

See Also

set_lower_bounds : Set parameter lower bounds

unconstrain_parameters : Remove all constraints of parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
set_upper_bounds(model, list('THETA(1)'=10))
model$parameters['THETA(1)']

## End(Not run)
```

set_weighted_error_model *set_weighted_error_model*

Description

Encode error model with one epsilon and W as weight

Usage

```
set_weighted_error_model(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to the same model

See Also

use_thetas_for_error_stdev : Use thetas to estimate error

Examples

```
## Not run:
model <- load_example_model("pheno")
set_weighted_error_model(model)

## End(Not run)
```

`set_zero_order_absorption` *set_zero_order_absorption*

Description

Set or change to zero order absorption rate.

Initial estimate for absorption rate is set the previous rate if available, otherwise it is set to the time of first observation/2.

Usage

```
set_zero_order_absorption(model)
```

Arguments

`model`	(Model) Model to set or change to first order absorption rate

Value

(Model) Reference to the same model

See Also

set_bolus_order_absorption

set_first_order_absorption

Examples

```
## Not run:
model <- load_example_model("pheno")
set_zero_order_absorption(model)
model$statements$ode_system

## End(Not run)
```

set_zero_order_elimination

set_zero_order_elimination

Description

Sets elimination to zero order.

Initial estimate for KM is set to 1% of smallest observation.

Usage

```
set_zero_order_elimination(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to same model

See Also

set_first_order_elimination

set_michaelis_menten_elimination

Examples

```
## Not run:
model <- load_example_model("pheno")
set_zero_order_elimination(model)
model$statements$ode_system

## End(Not run)
```

`simplify_expression` *simplify_expression*

Description

Simplify expression given constraints in model

Usage

```
simplify_expression(model, expr)
```

Arguments

`model`	(Model) Pharmpy model object
`expr`	(Expression) Expression to simplify

Value

(Expression) Simplified expression

Examples

```
## Not run:
conf$parameter_names <- c('comment', 'basic')
model <- load_example_model("pheno")
simplify_expression(model, "Abs(PTVCL)")
conf$parameter_names <- c('basic')

## End(Not run)
```

`solve_ode_system` *solve_ode_system*

Description

Replace ODE system with analytical solution if possible

Warnings This function can currently only handle the most simple of ODE systems.

Usage

```
solve_ode_system(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Model) Reference to the same pharmpy model object

Examples

```
## Not run:
model <- load_example_model("pheno")
model$statements$ode_system
solve_ode_system(model)

## End(Not run)
```

---

split_joint_distribution

split_joint_distribution

Description

Splits etas following a joint distribution into separate distributions.

Usage

```
split_joint_distribution(model, rvs = NULL)
```

Arguments

`model`	(Model) Pharmpy model
`rvs`	(str, vector) Name/names of etas to separate. If NULL, all etas that are IIVs and non-fixed will become single. NULL is default.

Value

(Model) Reference to the same model

See Also

create_joint_distribution : combine etas into a join distribution

Examples

```
## Not run:
model <- load_example_model("pheno")
create_joint_distribution(model, c('ETA(1)', 'ETA(2)'))
model$random_variables$etas
split_joint_distribution(model, c('ETA(1)', 'ETA(2)'))
model$random_variables$etas

## End(Not run)
```

`summarize_errors` *summarize_errors*

Description

Summarize errors and warnings from one or multiple model runs.

Summarize the errors and warnings found after running the model/models.

Usage

```
summarize_errors(models)
```

Arguments

`models`	(vector, Model) List of models or single model

Value

(data.frame) A DataFrame of errors with model name, category (error or warning), and an integer as index, an empty DataFrame if there were no errors or warnings found.

Examples

```
## Not run:
model <- load_example_model("pheno")
summarize_errors(model)

## End(Not run)
```

`summarize_individuals` *summarize_individuals*

Description

Creates a summary dataframe keyed by model-individual pairs for an input vector of models.

Content of the various columns:

+——————————-+———————————————————————————-+ | Column | Description | +=======================+== | `outlier_count` | Number of observations with CWRES > 5 | +——————————-+——-+ | `ofv` | Individual OFV | +——————————-+———————————————————————————-+ | `dofv_vs_parent` | Difference in individual OFV between this model and its parent model | +——————————-+————————————————————————————————-+ | `predicted_dofv` | Predicted dOFV if this individual was excluded | +——————————-+———————————————————————————-+ | `predicted_residual` | Predicted residual | +——————————-+———————————————————————————————-+

Usage

```
summarize_individuals(models)
```

Arguments

`models`	(vector of Models) Input models

Value

(data.frame | NULL) The summary as a dataframe

Examples

```
## Not run:
model <- load_example_model("pheno")
fit(model)
results <- run_tool(
    model=model,
    mfl='ABSORPTION(ZO);PERIPHERALS(c(1, 2))',
    algorithm='reduced_stepwise'
summarize_individuals([results$start_model, *results$models])

## End(Not run)
```

`summarize_individuals_count_table`
summarize_individuals_count_table

Description

Create a count table for individual data

Content of the various columns:

+——————————-+—————————————————————————————
—+ | Column | Description | +========================+==
| `inf_selection` | Number of subjects influential on model selection. ||| :math:`\mathrm{OFV}_{parent} - \mathrm{OFV} >`
||| :math:`\mathrm{OFV}_{parent} - \mathrm{iOFV}_{parent} - (\mathrm{OFV} - \mathrm{iOFV}) > 3.84`
| +——————————-+——————————————————————————————
——+ | `inf_params` | Number of subjects influential on parameters. predicted_dofv > 3.84 | +———
——————————-+——————————————————————————————+ |
`out_obs` | Number of subjects having at least one outlying observation (CWRES > 5) | +——————
—————-+——————————————————————————————+ | `out_ind`
| Number of outlying subjects. predicted_residual > 3.0 | +——————————-+——————
——————————————————————————+ | `inf_outlier` | Number of subjects both influential by any criteria and outlier by any criteria | +——————————-+——————
——————————————————————————+

Usage

```
summarize_individuals_count_table(models = NULL, df = NULL)
```

Arguments

`models`	(vector of models) List of models to summarize.
`df`	(data.frame) Output from a previous call to summarize_individuals.

Value

(data.frame) Table with one row per model.

See Also

summarize_individuals : Get raw individual data

`summarize_modelfit_results` *summarize_modelfit_results*

Description

Summarize results of model runs

Summarize different results after fitting a model, includes runtime, ofv, and parameter estimates (with errors). If include_all_estimation_steps is FALSE, only the last estimation step will be included (note that in that case, the minimization_successful value will be referring to the last estimation step, if last step is evaluation it will go backwards until it finds an estimation step that wasn't an evaluation).

Usage

```
summarize_modelfit_results(models, include_all_estimation_steps = FALSE)
```

Arguments

`models`	(vector, Model) List of models or single model
`include_all_estimation_steps`	(logical) Whether to include all estimation steps, default is FALSE

Value

(data.frame) A DataFrame of modelfit results with model name and estmation step as index.

Examples

```
## Not run:
model <- load_example_model("pheno")
summarize_modelfit_results(model)

## End(Not run)
```

---

transform_etas_boxcox *transform_etas_boxcox*

Description

Applies a boxcox transformation to selected etas

Initial estimate for lambda is 0.1 with bounds (-3, 3).

Usage

```
transform_etas_boxcox(model, list_of_etas = NULL)
```

Arguments

`model`	(Model) Pharmpy model to apply boxcox transformation to.
`list_of_etas`	(str, vector) Name/names of etas to transform. If NULL, all etas will be transformed (default).

Value

(Model) Reference to the same model

See Also

transform_etas_tdist

transform_etas_john_draper

Examples

```
## Not run:
model <- load_example_model("pheno")
transform_etas_boxcox(model, c("ETA(1)"))
model$statements$before_odes$full_expression("CL")

## End(Not run)
```

transform_etas_john_draper

transform_etas_john_draper

Description

Applies a John Draper transformation (1) to spelected etas

Initial estimate for lambda is 0.1 with bounds (-3, 3).

(1) John, J., Draper, N. (1980). An Alternative Family of Transformations. Journal of the Royal Statistical Society. Series C (Applied Statistics), 29(2), 190-197. doi:10.2307/2986305

Usage

```
transform_etas_john_draper(model, list_of_etas = NULL)
```

Arguments

`model`	(Model) Pharmpy model to apply John Draper transformation to.
`list_of_etas`	(str, vector) Name/names of etas to transform. If NULL, all etas will be transformed (default).

Value

(Model) Reference to the same model

See Also

transform_etas_boxcox

transform_etas_tdist

Examples

```
## Not run:
model <- load_example_model("pheno")
transform_etas_john_draper(model, c("ETA(1)"))
model$statements$before_odes$full_expression("CL")

## End(Not run)
```

`transform_etas_tdist` *transform_etas_tdist*

Description

Applies a t-distribution transformation to selected etas

Initial estimate for degrees of freedom is 80 with bounds (3, 100).

Usage

```
transform_etas_tdist(model, list_of_etas = NULL)
```

Arguments

`model`	(Model) Pharmpy model to apply t distribution transformation to.
`list_of_etas`	(str, vector) Name/names of etas to transform. If NULL, all etas will be transformed (default).

Value

(Model) Reference to the same model

See Also

transform_etas_boxcox

transform_etas_john_draper

Examples

```
## Not run:
model <- load_example_model("pheno")
transform_etas_tdist(model, c("ETA(1)"))
model$statements$before_odes$full_expression("CL")

## End(Not run)
```

`translate_nmtran_time` *translate_nmtran_time*

Description

Translate NM-TRAN TIME and DATE column into one TIME column

If dataset of model have special NM-TRAN TIME and DATE columns these will be translated into one single time column with time in hours.

Warnings Use this function with caution. For example reset events are currently not taken into account.

Usage

```
translate_nmtran_time(model)
```

Arguments

`model`	(Model) Pharmpy model object

Value

(Model) Reference to the same model object

`unconstrain_parameters` *unconstrain_parameters*

Description

Remove all constraints from parameters

Usage

```
unconstrain_parameters(model, parameter_names)
```

Arguments

`model`	(Model) Pharmpy model
`parameter_names`	(vector) Remove all constraints for the listed parameters

Value

(Model) Reference to the same model object

See Also

set_lower_bounds : Set parameter lower bounds

set_upper_bounds : Set parameter upper bounds

unfix_parameters : Unfix parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
model$parameters['THETA(1)']
unconstrain_parameters(model, c('THETA(1)'))
model$parameters['THETA(1)']

## End(Not run)
```

`undrop_columns` *undrop_columns*

Description

Undrop columns of model

Usage

```
undrop_columns(model, column_names)
```

Arguments

`model`	(Model) Pharmpy model object
`column_names`	(vector or str) List of column names or one column name to undrop

Value

(Model) Reference to same model object

See Also

drop_dropped_columns : Drop all columns marked as drop

drop_columns : Drop or mark columns as dropped

Examples

```
## Not run:
model <- load_example_model("pheno")
drop_columns(model, c('WGT', 'APGR'), mark=TRUE)
undrop_columns(model, 'WGT')

## End(Not run)
```

unfix_parameters *unfix_parameters*

Description

Unfix parameters

Unfix all listed parameters

Usage

```
unfix_parameters(model, parameter_names)
```

Arguments

`model`	(Model) Pharmpy model
`parameter_names`	(vector or str) one parameter name or a vector of parameter names

Value

(Model) Reference to the same model object

See Also

unfix_paramaters_to : Unfixing parameters and setting a new initial estimate in the same function

fix_parameters : Fix parameters

fix_or_unfix_parameters : Fix or unfix parameters (given boolean)

fix_parameters_to : Fixing and setting parameter initial estimates in the same function

unconstrain_parameters : Remove all constraints of parameters

Examples

```
## Not run:
model <- load_example_model("pheno")
fix_parameters(model, c('THETA(1)', 'THETA(2)', 'THETA(3)'))
model$parameters$fix
unfix_parameters(model, 'THETA(1)')
model$parameters$fix

## End(Not run)
```

`unfix_parameters_to` *unfix_parameters_to*

Description

Unfix parameters to

Unfix all listed parameters to specified value/values

Usage

```
unfix_parameters_to(model, inits)
```

Arguments

`model`	(Model) Pharmpy model
`inits`	(list) Inits for all parameters to unfix and change init

Value

(Model) Reference to the same model object

See Also

fix_parameters : Fix parameters

fix_or_unfix_parameters : Fix or unfix parameters (given boolean)

unfix_paramaters : Unfixing parameters

fix_paramaters_to : Fixing parameters and setting a new initial estimate in the same

function

Examples

```
## Not run:
model <- load_example_model("pheno")
fix_parameters(model, c('THETA(1)', 'THETA(2)', 'THETA(3)'))
model$parameters$fix
unfix_parameters_to(model, {'THETA(1)': 0.5})
model$parameters$fix
model$parameters['THETA(1)']

## End(Not run)
```

update_inits *update_inits*

Description

Update initial parameter estimate for a model

Updates initial estimates of population parameters for a model from its modelfit_results. If the model has used initial estimates for individual estimates these will also be updated. If the new initial estimates are out of bounds or NaN this function will raise.

Usage

```
update_inits(
  model,
  force_individual_estimates = FALSE,
  move_est_close_to_bounds = FALSE
)
```

Arguments

`model`	(Model) Pharmpy model to update initial estimates
`force_individual_estimates`	(logical) Update initial individual estimates even if model din't use them previously.
`move_est_close_to_bounds`	(logical) Move estimates that are close to bounds. If correlation >0.99 the correlation will be set to 0.9, if variance is <0.001 the variance will be set to 0.01.

Value

(Model) Reference to the same model

Examples

```
## Not run:
model <- load_example_model("pheno")   # This model was previously fitted to its data
model$parameters$inits
update_inits(model)
model$parameters$inits

## End(Not run)
```

use_thetas_for_error_stdev *use_thetas_for_error_stdev*

Description

Use thetas to estimate standard deviation of error

Usage

```
use_thetas_for_error_stdev(model)
```

Arguments

`model`	(Model) Pharmpy model

Value

(Model) Reference to the same model

See Also

set_weighted_error_model : Encode error model with one epsilon and weight

Examples

```
## Not run:
model <- load_example_model("pheno")
use_thetas_for_error_stdev(model)
model$statements$find_assignment("Y")

## End(Not run)
```

write_csv *write_csv*

Description

Write dataset to a csv file

Usage

```
write_csv(model, path = NULL, force = FALSE)
```

Arguments

`model`	(Model) Model whose dataset to write to file
`path`	(Path) Destination path. Default is to use original path with .csv suffix.
`force`	(logical) Overwrite file with same path. Default is FALSE.

Value

(Path) path to the written file.

Examples

```
## Not run:
model <- load_example_model("pheno")
write_csv(model, path="newdataset$csv")

## End(Not run)
```

`write_model` *write_model*

Description

Write model code to file

Usage

```
write_model(model, path = "", force = TRUE)
```

Arguments

`model`	(Model) Pharmpy model
`path`	(str) Destination path
`force`	(logical) Force overwrite, default is TRUE

Value

(Model) Reference to the same model object

Examples

```
## Not run:
model <- load_example_model("pheno")
write_model(model)

## End(Not run)
```

`write_results` *write_results*

Description

Write results object to json (or csv) file

Note that the csv-file cannot be read into a results object again.

Usage

```
write_results(results, path, lzma = FALSE, csv = FALSE)
```

Arguments

`results`	(Results) Pharmpy results object
`path`	(Path) Path to results file
`lzma`	(logical) TRUE for lzma compression. Not applicable to csv file
`csv`	(logical) Save as csv file

Index