Package ‘PRA’

April 8, 2026

Type Package

Title Project Risk Analysis

Version 0.4.0

Description Data analysis for Project Risk Management via the Second Moment Method, Monte Carlo Simulation, Contingency Analysis, Sensitivity Analysis, Earned Value Management, Learning Curves, Bayesian Methods, and more.

Depends R (>= 4.1.0)

Imports mc2d, minpack.lm, stats

Encoding UTF-8

URL https://paulgovan.github.io/PRA/, https://github.com/paulgovan/PRA

BugReports https://github.com/paulgovan/PRA/issues

Suggests base64enc, bslib, ellmer, ggplot2, htmltools, jsonlite, knitr, ragnar, rmarkdown, scales, shiny, shinychat, mockr, testthat (>= 3.0.0), vitals, withr

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author Paul Govan [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-1821-8492>)

Maintainer Paul Govan <paul.govan2@gmail.com>

Repository CRAN

Date/Publication 2026-04-08 19:10:33 UTC

Contents

`ac` *Actual Cost (AC).*

Description

Calculates the Actual Cost (AC) of work completed based on the actual costs incurred at each time period.

Usage

```
ac(actual_costs, time_period, cumulative = TRUE)
```

Arguments

`actual_costs`	Vector of actual costs incurred at each time period. Can be either period costs (cost per period) or cumulative costs depending on the cumulative parameter.
`time_period`	Current time period.
`cumulative`	Logical. If TRUE (default), actual_costs are already cumulative and the value at time_period is returned directly. If FALSE, actual_costs are period costs and will be summed up to time_period.

Value

The function returns the Actual Cost (AC) of work completed to date.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

`pv`, `ev`, `sv`, `cv`, `spi`, `cpi`, `eac`

Examples

```
# Using cumulative costs (default)
cumulative_costs <- c(9000, 27000, 63000, 133000, 233000)
time_period <- 3
ac <- ac(cumulative_costs, time_period)
cat("Actual Cost (AC):", ac, "\n")

# Using period costs
period_costs <- c(9000, 18000, 36000, 70000, 100000)
ac <- ac(period_costs, time_period, cumulative = FALSE)
cat("Actual Cost (AC):", ac, "\n")
```

`add_documents` *Add Custom Documents to the PRA Knowledge Base*

Description

Ingests additional markdown or text documents into an existing PRA knowledge base. Use this to extend the agent's knowledge with your own project documentation, standards, templates, or lessons learned.

Usage

```
add_documents(store, path, embed_model = "nomic-embed-text")
```

Arguments

store	A ragnar store object from build_knowledge_base().
path	Character. Path to a file or directory. If a directory, all .md and .txt files are ingested recursively.
embed_model	Character. Ollama embedding model (default "nomic-embed-text").

Value

The store object (invisibly), updated with the new documents.

Examples

```
## Not run:
store <- build_knowledge_base()

# Add a single file
add_documents(store, "path/to/my_risk_register.md")

# Add all .md and .txt files in a directory
add_documents(store, "path/to/project_docs/")

## End(Not run)
```

---

build_knowledge_base *Build the PRA Knowledge Base for RAG Retrieval*

Description

Reads the curated risk analysis knowledge files bundled with the PRA package, chunks them, generates embeddings via Ollama, and stores them in a DuckDB-backed ragnar knowledge base for retrieval-augmented generation.

Usage

```
build_knowledge_base(
  store_path = NULL,
  embed_model = "nomic-embed-text",
  overwrite = FALSE
)
```

Arguments

store_path	Path to store the DuckDB knowledge base. Defaults to a cache directory under tools::R_user_dir().
embed_model	Ollama embedding model name (default "nomic-embed-text").
overwrite	Logical. If TRUE, rebuild the knowledge base even if a cached version exists. Default FALSE.

Details

The knowledge base is built once and cached to disk. Subsequent calls with the same `store_path` load the existing store.

Value

A ragnar store object that can be passed to `retrieve_context()`.

Examples

```
## Not run:
store <- build_knowledge_base()
context <- retrieve_context(store, "How do I run a Monte Carlo simulation?")

## End(Not run)
```

`contingency` *Contingency Calculation.*

Description

This function calculates the contingency required for a project based on the results of a Monte Carlo simulation. The contingency is determined by the difference between the specified high percentile (phigh) and the base percentile (pbase) of the total project duration distribution.

Usage

```
contingency(sims, phigh = 0.95, pbase = 0.5)
```

Arguments

`sims`	List of results from a Monte Carlo simulation containing the total project duration distribution.
`phigh`	Percentile level for contingency calculation. Default is 0.95 (95th percentile).
`pbase`	Base level for contingency calculation. Default is 0.5 (50th percentile).

Value

The function returns the value of calculated contingency based on the specified percentiles.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Set the number os simulations and the task distributions for a toy project.
num_sims <- 10000
task_dists <- list(
  list(type = "normal", mean = 10, sd = 2), # Task A: Normal distribution
  list(type = "triangular", a = 5, b = 10, c = 15), # Task B: Triangular distribution
  list(type = "uniform", min = 8, max = 12) # Task C: Uniform distribution
)

# Set the correlation matrix for the correlations between tasks.
cor_mat <- matrix(c(
  1, 0.5, 0.3,
  0.5, 1, 0.4,
  0.3, 0.4, 1
), nrow = 3, byrow = TRUE)

# Run the Monte Carlo simulation.
results <- mcs(num_sims, task_dists, cor_mat)

# Calculate the contingency and print the results.
contingency <- contingency(results, phigh = 0.95, pbase = 0.50)
cat("Contingency based on 95th percentile and 50th percentile:", contingency)

# Without correlation matrix
results_indep <- mcs(num_sims, task_dists)
contingency_indep <- contingency(results_indep,
  phigh = 0.95,
  pbase = 0.50
)
cat("Contingency based on 95th percentile and 50th percentile (
independent tasks):", contingency_indep)

# Build a barplot to visualize the contingency results.
contingency_data <- data.frame(
  Scenario = c("With Correlation", "Independent Tasks"),
  Contingency = c(contingency, contingency_indep)
)
barplot(
  height = contingency_data$Contingency,
  names = contingency_data$Scenario,
  col = c("orange", "purple"),
  horiz = TRUE,
  xlab = "Contingency",
  ylab = "Scenario"
)
title("Contingency Calculation for Project Scenarios")
```

`cor_matrix` *Generate Correlation Matrix from Random Samples.*

Description

This function generates random samples from specified probability distributions and computes the correlation matrix for the generated samples.

Usage

```
cor_matrix(num_samples = 100, num_vars = 5, dists)
```

Arguments

num_samples	The number of samples to generate.
num_vars	The number of distributions to sample.
dists	A list describing each distribution. Each element should be a function that generates random samples. The names of the list elements will be used to identify the distributions.

Value

The function returns the correlation matrix for the distributions.

References

Govan, Paul, and Ivan Damnjanovic. "The resource-based view on project risk management." Journal of construction engineering and management 142.9 (2016): 04016034.

Examples

```
# List of probability distributions
dists <- list(
  normal = function(n) rnorm(n, mean = 0, sd = 1),
  uniform = function(n) runif(n, min = 0, max = 1),
  exponential = function(n) rexp(n, rate = 1),
  poisson = function(n) rpois(n, lambda = 1),
  binomial = function(n) rbinom(n, size = 10, prob = 0.5)
)

# Generate correlation matrix
cor_matrix <- cor_matrix(num_samples = 100, num_vars = 5, dists = dists)

# Print correlation matrix
print(cor_matrix)
```

cost_pdf *Cost Probability Density.*

Description

This function generates random samples from a mixture model representing the cost 'A' associated with multiple risk events 'R_i'. Each risk event has its own probability, mean, and standard deviation for the cost distribution. The function also accounts for a baseline cost when no risk event occurs.

Usage

```
cost_pdf(
  num_sims,
  risk_probs,
  means_given_risks,
  sds_given_risks,
  base_cost = 0
)
```

Arguments

`num_sims` Number of random samples to draw from the mixture model.

`risk_probs` A vector of probabilities for each risk event 'R_i'.

`means_given_risks` A vector of means of the normal distribution for cost 'A' given each risk event 'R_i'.

`sds_given_risks` A vector of standard deviations of the normal distribution for cost 'A' given each risk event 'R_i'.

`base_cost` The baseline cost given no risk event occurs.

Value

A numeric vector of random samples from the mixture model.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Example with three risk events
num_sims <- 1000
risk_probs <- c(0.3, 0.5, 0.2)
means_given_risks <- c(10000, 15000, 5000)
```

```
sds_given_risks <- c(2000, 1000, 1000)
base_cost <- 2000
samples <- cost_pdf(
  num_sims = num_sims,
  risk_probs = risk_probs,
  means_given_risks = means_given_risks,
  sds_given_risks = sds_given_risks,
  base_cost = base_cost
)
hist(samples, breaks = 30, col = "skyblue", main = "Histogram of Cost", xlab = "Cost")
```

---

cost_post_pdf *Posterior Cost Probability Density.*

Description

This function generates random samples from the posterior distribution of the cost 'A' given observations of multiple risk events 'R_i'. Each risk event has its own mean and standard deviation for the cost distribution. The function also accounts for a baseline cost when no risk event occurs.

Usage

```
cost_post_pdf(
  num_sims,
  observed_risks,
  means_given_risks,
  sds_given_risks,
  base_cost = 0
)
```

Arguments

`num_sims`	Number of random samples to draw from the posterior distribution.
`observed_risks`	A vector of observed values for each risk event 'R_i' (1 if observed, 0 if not observed, NA if unobserved).
`means_given_risks`	A vector of means of the normal distribution for cost 'A' given each risk event 'R_i'.
`sds_given_risks`	A vector of standard deviations of the normal distribution for cost 'A' given each risk event 'R_i'.
`base_cost`	The baseline cost given no risk event occurs.

Value

A numeric vector of random samples from the posterior distribution of costs.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Example with three risk events
num_sims <- 1000
observed_risks <- c(1, NA, 1)
means_given_risks <- c(10000, 15000, 5000)
sds_given_risks <- c(2000, 1000, 1000)
base_cost <- 2000
posterior_samples <- cost_post_pdf(
  num_sims = num_sims,
  observed_risks = observed_risks,
  means_given_risks = means_given_risks,
  sds_given_risks = sds_given_risks,
  base_cost = base_cost
)
hist(posterior_samples, breaks = 30, col = "skyblue", main = "Posterior Cost PDF", xlab = "Cost")
```

---

cpi *Cost Performance Index (CPI).*

Description

Calculates the Cost Performance Index (CPI) of work completed based on the Earned Value (EV) and Actual Cost (AC).

Usage

```
cpi(ev, ac)
```

Arguments

ev	Earned Value.
ac	Actual Cost.

Value

The function returns the Cost Performance Index (CPI) of work completed.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

pv, ev, ac, sv, cv, spi, eac

Examples

```
# Set the BAC and actual % complete for an example project.
bac <- 100000
actual_per_complete <- 0.35

# Calcualte the EV
ev <- ev(bac, actual_per_complete)

# Set the actual costs and current time period and calculate the AC.
actual_costs <- c(9000, 18000, 36000, 70000, 100000)
time_period <- 3
ac <- ac(actual_costs, time_period)

# Calculate the CPI and print the results.
cpi <- cpi(ev, ac)
cat("Cost Performance Index (CPI):", cpi, "\n")
```

---

cv *Cost Variance (CV).*

Description

Calculates the Cost Variance (CV) of work completed based on the Earned Value (EV) and Actual Cost (AC).

Usage

```
cv(ev, ac)
```

Arguments

ev	Earned Value.
ac	Actual Cost.

Value

The function returns the Cost Variance (CV) of work completed.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

pv, ev, ac, sv, spi, cpi, eac

Examples

```
# Set the BAC and actual % complete for an example project.
bac <- 100000
actual_per_complete <- 0.35

# Calcualte the EV
ev <- ev(bac, actual_per_complete)

# Set the actual costs and current time period and calculate the AC.
actual_costs <- c(9000, 18000, 36000, 70000, 100000)
time_period <- 3
ac <- ac(actual_costs, time_period)

# Calculate the CV and print the results.
cv <- cv(ev, ac)
cat("Cost Variance (CV):", cv, "\n")
```

---

eac *Estimate at Completion (EAC).*

Description

Calculates the Estimate at Completion (EAC) using various methods based on project performance assumptions.

Usage

```
eac(bac, method = "typical", cpi = NULL, ac = NULL, ev = NULL, spi = NULL)
```

Arguments

bac	Budget at Completion (BAC) (total planned budget).
method	The EAC calculation method. One of: • "typical" (default): EAC = BAC / CPI. Assumes future work performed at current cost efficiency. • "atypical": EAC = AC + (BAC - EV). Assumes future work performed at planned rate. • "combined": EAC = AC + (BAC - EV) / (CPI * SPI). Considers both cost and schedule performance.
cpi	Cost Performance Index (CPI). Required for "typical" and "combined" methods.
ac	Actual Cost. Required for "atypical" and "combined" methods.
ev	Earned Value. Required for "atypical" and "combined" methods.
spi	Schedule Performance Index. Required for "combined" method.

Value

The function returns the Estimate at Completion (EAC).

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

pv, ev, ac, sv, cv, spi, cpi, etc, vac, tcpi

Examples

```
# Method 1: Typical - assumes current CPI continues
bac <- 100000
cpi <- 0.83
eac <- eac(bac, cpi = cpi)
cat("EAC (typical):", round(eac, 2), "\n")

# Method 2: Atypical - assumes future work at planned rate
ac <- 63000
ev <- 35000
eac <- eac(bac, method = "atypical", ac = ac, ev = ev)
cat("EAC (atypical):", round(eac, 2), "\n")

# Method 3: Combined - considers both CPI and SPI
spi <- 0.875
eac <- eac(bac, method = "combined", cpi = cpi, ac = ac, ev = ev, spi = spi)
cat("EAC (combined):", round(eac, 2), "\n")
```

---

etc — *Estimate to Complete (ETC).*

Description

Calculates the Estimate to Complete (ETC), which is the expected cost to finish the remaining work.

Usage

```
etc(bac, ev, cpi = NULL)
```

Arguments

bac	Budget at Completion (BAC) (total planned budget).
ev	Earned Value.
cpi	Cost Performance Index. If NULL, assumes remaining work will be completed at planned cost (ETC = BAC - EV). If provided, adjusts for current performance (ETC = (BAC - EV) / CPI).

Value

The function returns the Estimate to Complete (ETC).

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

eac, vac, tcpi

Examples

```
bac <- 100000
ev <- 35000
cpi <- 0.83

# ETC assuming remaining work at planned rate
etc <- etc(bac, ev)
cat("ETC (planned rate):", etc, "\n")

# ETC assuming remaining work at current CPI
etc <- etc(bac, ev, cpi)
cat("ETC (current CPI):", round(etc, 2), "\n")
```

---

ev — *Earned Value (EV).*

Description

Calculates the Earned Value (EV) of work completed based on the Budget at Completion (BAC) and the actual work completion percentage.

Usage

```
ev(bac, actual_per_complete)
```

Arguments

bac	Budget at Completion (BAC) (total planned budget).
actual_per_complete	Actual work completion percentage.

Value

The function returns the Earned Value (EV) of work completed.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

pv, ac, sv, cv, spi, cpi, eac

Examples

```
# Set the BAC and actual % complete for a toy project.
bac <- 100000
actual_per_complete <- 0.35

# Calculate the EV and print the results.
ev <- ev(bac, actual_per_complete)
cat("Earned Value (EV):", ev, "\n")
```

---

fit_sigmoidal   *Fit a Sigmoidal Model to Data.*

Description

This function fits a sigmoidal model (Pearl, Gompertz, or Logistic) to the provided data.

Usage

```
fit_sigmoidal(data, x_col, y_col, model_type)
```

Arguments

data	A data frame containing the time (x_col) and completion (y_col) vectors.
x_col	The name of the time vector.
y_col	The name of the completion vector.
model_type	The name of the sigmoidal model (Pearl, Gompertz, or Logistic).

Value

The function returns a list of results for the sigmoidal model.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Set up a data frame of time and completion percentage data
data <- data.frame(time = 1:10, completion = c(5, 15, 40, 60, 70, 75, 80, 85, 90, 95))

# Fit a logistic model to the data.
fit <- fit_sigmoidal(data, "time", "completion", "logistic")

# Use the model to predict future completion times.
predictions <- predict_sigmoidal(fit, seq(min(data$time), max(data$time),
  length.out = 100
), "logistic")

# Predict with 95% confidence bounds
predictions_ci <- predict_sigmoidal(fit, seq(min(data$time), max(data$time),
  length.out = 100
), "logistic", conf_level = 0.95)
```

---

grandparent_dsm *Risk-based 'Grandparent' Design Structure Matrix (DSM).*

Description

This function computes the Risk-based 'Grandparent' Design Structure Matrix (DSM) from given Resource-Task Matrix 'S' and Risk-Resource Matrix 'R'. The 'Grandparent' DSM indicates the number of risks shared between each pair of tasks in a project.

Usage

```
grandparent_dsm(S, R)
```

Arguments

S	Resource-Task Matrix 'S' giving the links (arcs) between resources and tasks. Rows represent resources and columns represent tasks.
R	Risk-Resource Matrix 'R' giving the links (arcs) between risks and resources. Rows represent risks and columns represent resources.

Value

An S3 object of class `"dsm"` with the following components:

matrix The Risk-based 'Grandparent' DSM giving the number of risks shared between each task.

type Character string `"grandparent"`.

n_tasks Number of tasks (columns in S).

n_resources Number of resources (rows in S).

n_risks Number of risks (rows in R).

References

Govan, Paul, and Ivan Damnjanovic. "The resource-based view on project risk management." Journal of construction engineering and management 142.9 (2016): 04016034.

Examples

```
# Set the S and R matrices and print the results.
S <- matrix(c(1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1), nrow = 3, ncol = 4)
R <- matrix(c(1, 1, 0, 1, 0, 0), nrow = 2, ncol = 3)
cat("Resource-Task Matrix (3 resources x 4 tasks):\n")
print(S)
cat("\nRisk-Resource Matrix (2 risks x 3 resources):\n")
print(R)
# Calculate the Risk-based Grandparent Matrix and print the results.
risk_dsm <- grandparent_dsm(S, R)
print(risk_dsm)
```

`mcs` *Monte Carlo Simulation.*

Description

This function performs a Monte Carlo simulation to estimate the total duration of a project based on individual task distributions and an optional correlation matrix.

Usage

```
mcs(num_sims, task_dists, cor_mat = NULL)
```

Arguments

`num_sims`	The number of simulations to run.
`task_dists`	A list of lists describing each task distribution with its parameters. Each task distribution should be specified as a list with a "type" field (indicating the distribution type: "normal", "triangular", or "uniform") and the corresponding parameters: for "normal" (mean, sd), for "triangular" (a, b, c), and for "uniform" (min, max). For example: list(list(type = "normal", mean = 10, sd = 2), list(type = "triangular", a = 5, b = 10, c = 15), list(type = "uniform", min = 8, max = 12))
`cor_mat`	The correlation matrix for the tasks (Optional). If not provided, tasks are assumed to be independent.

Value

The function returns a list of the total mean, variance, standard deviation, and percentiles for the project.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Set the number of simulations and task distributions for a toy project.
num_sims <- 10000
task_dists <- list(
  list(type = "normal", mean = 10, sd = 2), # Task A: Normal distribution
  list(type = "triangular", a = 5, b = 10, c = 15), # Task B: Triangular distribution
  list(type = "uniform", min = 8, max = 12) # Task C: Uniform distribution
)

# Set the correlation matrix for the correlations between tasks.
cor_mat <- matrix(c(
  1, 0.5, 0.3,
  0.5, 1, 0.4,
  0.3, 0.4, 1
), nrow = 3, byrow = TRUE)

# Run the Monte Carlo sumulation and print the results.
results <- mcs(num_sims, task_dists, cor_mat)
cat("Mean Total Duration:", results$total_mean, "\n")
cat("Variance of Total Variance:", results$total_variance, "\n")
cat("Standard Deviation of Total Duration:", results$total_sd, "\n")
cat("5th Percentile:", results$percentiles[1], "\n")
cat("Median (50th Percentile):", results$percentiles[2], "\n")
cat("95th Percentile:", results$percentiles[3], "\n")
hist(results$total_distribution,
  breaks = 50, main = "Distribution of Total Project Duration",
  xlab = "Total Duration", col = "skyblue", border = "white"
)
legend("topright", legend = c("Total Duration Distribution"), fill = c("skyblue"))
```

---

parent_dsm *Resource-based 'Parent' Design Structure Matrix (DSM).*

Description

This function computes the Resource-based 'Parent' Design Structure Matrix (DSM) from a given Resource-Task Matrix 'S'. The 'Parent' DSM indicates the number of resources shared between each pair of tasks in a project.

Usage

```
parent_dsm(S)
```

Arguments

S   Resource-Task Matrix 'S' giving the links (arcs) between resources and tasks. Rows represent resources and columns represent tasks.

Value

An S3 object of class "dsm" with the following components:

matrix The Resource-based 'Parent' DSM giving the number of resources shared between each task.

type Character string "parent".

n_tasks Number of tasks (columns in S).

n_resources Number of resources (rows in S).

References

Govan, Paul, and Ivan Damnjanovic. "The resource-based view on project risk management." Journal of construction engineering and management 142.9 (2016): 04016034.

Examples

```
# Set the S matrix for a toy project (3 resources x 4 tasks).
s <- matrix(c(1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1), nrow = 3, ncol = 4)
cat("Resource-Task Matrix:\n")
print(s)

# Calculate the Resource-based Parent DSM and print the results.
resource_dsm <- parent_dsm(s)
print(resource_dsm)
```

---

plot.dsm   *Plot a DSM heatmap.*

Description

Displays the Design Structure Matrix as a heatmap where color intensity represents the number of shared resources (parent) or risks (grandparent) between task pairs.

Usage

```
## S3 method for class 'dsm'
plot(x, main = NULL, col = NULL, ...)
```

Arguments

x	A dsm object returned by parent_dsm() or grandparent_dsm().
main	Optional plot title. If NULL, a default title is generated.
col	Color palette vector. If NULL, uses grDevices::heat.colors().
...	Additional arguments passed to graphics::image().

Value

Invisibly returns x.

---

plot_sigmoidal *Plot a Fitted Sigmoidal Model.*

Description

This function creates a base R plot of a fitted sigmoidal model with the original data points, fitted curve, and optional confidence bounds.

Usage

```
plot_sigmoidal(
  fit,
  data,
  x_col,
  y_col,
  model_type,
  conf_level = NULL,
  n_points = 100,
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  line_col = "red",
  ci_col = "lightblue",
  pch = 16,
  ...
)
```

Arguments

fit	A fitted sigmoidal model object from fit_sigmoidal.
data	The original data frame used to fit the model.
x_col	The name of the x (time) column in the data.
y_col	The name of the y (completion) column in the data.
model_type	The type of model (pearl, gompertz, or logistic).

conf_level	Optional confidence level for confidence bounds (e.g., 0.95 for 95%). If NULL (default), no confidence bounds are plotted.
n_points	Number of points to use for the fitted curve (default 100).
main	Plot title. If NULL, a default title is generated.
xlab	X-axis label. If NULL, uses x_col.
ylab	Y-axis label. If NULL, uses y_col.
line_col	Color for the fitted curve (default "red").
ci_col	Color for the confidence band (default "lightblue").
pch	Point character for data points (default 16).
...	Additional arguments passed to plot().

Value

Invisibly returns the predictions data frame.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Set up a data frame of time and completion percentage data
data <- data.frame(time = 1:10, completion = c(5, 15, 40, 60, 70, 75, 80, 85, 90, 95))

# Fit a logistic model to the data.
fit <- fit_sigmoidal(data, "time", "completion", "logistic")

# Plot the fitted model
plot_sigmoidal(fit, data, "time", "completion", "logistic")

# Plot with 95% confidence bounds
plot_sigmoidal(fit, data, "time", "completion", "logistic", conf_level = 0.95)

# Customize the plot
plot_sigmoidal(fit, data, "time", "completion", "logistic",
  conf_level = 0.95,
  main = "Project Completion Forecast",
  xlab = "Time (weeks)",
  ylab = "Completion (%)",
  line_col = "blue",
  ci_col = "lightgray"
)
```

pra_app *Launch the PRA Risk Analysis Agent App*

Description

Starts a Shiny app that provides an interactive chat interface to the PRA risk analysis agent. The agent can select and execute PRA tools (Monte Carlo simulation, EVM, Bayesian inference, etc.) in response to natural language questions. Uses shinychat for a polished streaming chat experience with inline tool result display.

Usage

```
pra_app(
  model = "llama3.2",
  rag = TRUE,
  embed_model = "nomic-embed-text",
  port = NULL,
  launch.browser = TRUE
)
```

Arguments

`model`	Character. Ollama model name (default `"llama3.2"`).
`rag`	Logical. Whether to enable RAG context retrieval (default `TRUE`).
`embed_model`	Character. Ollama embedding model for RAG (default `"nomic-embed-text"`).
`port`	Integer. Port for the Shiny app (default `NULL` lets Shiny choose).
`launch.browser`	Logical. Whether to open a browser (default `TRUE`).

Details

Requires Ollama to be running locally with the specified model downloaded.

Value

None. This function is called to launch the shiny app.

Examples

```
## Not run:
# Ensure Ollama is running, then:
pra_app()

# With a specific model:
pra_app(model = "qwen2.5")

## End(Not run)
```

pra_chat   *Create a PRA Risk Analysis Chat Agent*

Description

Creates an ellmer chat object configured as a project risk analysis expert, with all PRA functions registered as tools and optional RAG context retrieval from the bundled knowledge base.

Usage

```
pra_chat(
  chat = NULL,
  model = "llama3.2",
  rag = TRUE,
  embed_model = .pra_default_embed_model
)
```

Arguments

`chat`	An optional pre-configured ellmer chat object. If provided, `model` is ignored and tools are registered on this object instead.
`model`	Character. Ollama model name (default `"llama3.2"`). Must support tool calling. Other options: `"qwen2.5"`, `"llama3.1:70b"`.
`rag`	Logical. Whether to use RAG context from the PRA knowledge base (default `TRUE`). Requires Ollama embedding model to be available.
`embed_model`	Character. Ollama embedding model for RAG (default `"nomic-embed-text"`). Only used when `rag = TRUE`.

Details

By default, uses a local Ollama model for fully offline, private operation. Alternatively, supply a pre-configured ellmer chat object (e.g., `ellmer::chat_openai()`) via the `chat` parameter for cloud-hosted models.

Value

A configured ellmer chat object with PRA tools registered. Use `chat$chat("your question")` to interact.

Examples

```
## Not run:
# Default: local Ollama model
chat <- pra_chat()
chat$chat("Run a Monte Carlo simulation for a 3-task project with
  Task A: normal(10, 2), Task B: triangular(5, 10, 15), Task C: uniform(8, 12)")
```

```
# Use a cloud model for better accuracy
chat <- pra_chat(chat = ellmer::chat_openai(model = "gpt-4o"))

# Follow-up questions use conversation context
chat$chat("What is the contingency reserve at 95% confidence?")

## End(Not run)
```

pra_tools *Create PRA Tool Definitions for LLM Agent*

Description

Creates a list of ellmer tool objects that wrap PRA's exported functions for use with an LLM agent. Each tool includes a description that helps the LLM select the appropriate analysis method and properly format parameters.

Usage

```
pra_tools()
```

Details

Tool wrappers handle serialization between the LLM (JSON strings) and R (lists, matrices, data.frames). Complex inputs like task distribution lists and correlation matrices are accepted as JSON strings and deserialized internally.

Large output vectors (e.g., Monte Carlo simulation samples) are summarized to mean, sd, and key percentiles rather than returning the full vector to the LLM. The full results are stored in the package environment for use by downstream tools (e.g., contingency analysis needs the full distribution).

When used with shinychat, tool results include rich HTML display with inline plots via `ellmer::ContentToolResult`.

Value

A list of ellmer tool objects.

Examples

```
## Not run:
tools <- pra_tools()
chat <- ellmer::chat_ollama(model = "llama3.2")
for (tool in tools) chat$register_tool(tool)

## End(Not run)
```

`predict_sigmoidal` *Predict a Sigmoidal Function Using Fitted Model.*

Description

This function predicts values using a fitted sigmoidal model (Pearl, Gompertz, or Logistic) over a specified range of time values.

Usage

```
predict_sigmoidal(fit, x_range, model_type, conf_level = NULL)
```

Arguments

`fit`	A list containing the results of a sigmoidal model.
`x_range`	A vector of time values for the prediction.
`model_type`	The type of model (Pearl, Gompertz, or Logistic) for the prediction.
`conf_level`	Optional confidence level for confidence bounds (e.g., 0.95 for 95%). If NULL (default), no confidence bounds are computed.

Value

The function returns a data frame containing the time (x), predicted values (pred), and optionally lower (lwr) and upper (upr) confidence bounds.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Set up a data frame of time and completion percentage data
data <- data.frame(time = 1:10, completion = c(5, 15, 40, 60, 70, 75, 80, 85, 90, 95))

# Fit a logistic model to the data.
fit <- fit_sigmoidal(data, "time", "completion", "logistic")

# Use the model to predict future completion times.
predictions <- predict_sigmoidal(fit, seq(min(data$time), max(data$time),
  length.out = 100
), "logistic")

# Predict with 95% confidence bounds
predictions_ci <- predict_sigmoidal(fit, seq(min(data$time), max(data$time),
  length.out = 100
), "logistic", conf_level = 0.95)
```

print.dsm *Print a DSM object.*

Description

Print a DSM object.

Usage

```
## S3 method for class 'dsm'
print(x, ...)
```

Arguments

x	A dsm object returned by parent_dsm() or grandparent_dsm().
...	Additional arguments passed to print.default().

Value

Invisibly returns x.

print.mcs *Print method for Monte Carlo Simulation results.*

Description

Displays the total mean, variance, standard deviation, and percentiles of the Monte Carlo Simulation results in a readable format.

Usage

```
## S3 method for class 'mcs'
print(x, ...)
```

Arguments

x	An object of class "mcs".
...	Additional arguments (not used).

Value

None. Prints the results to the console.

Examples

```
# Set the number of simulations and task distributions for a toy project.
num_sims <- 10000
task_dists <- list(
  list(type = "normal", mean = 10, sd = 2), # Task A: Normal distribution
  list(type = "triangular", a = 5, b = 10, c = 15), # Task B: Triangular distribution
  list(type = "uniform", min = 8, max = 12) # Task C: Uniform distribution
)

# Set the correlation matrix for the correlations between tasks.
cor_mat <- matrix(c(
  1, 0.5, 0.3,
  0.5, 1, 0.4,
  0.3, 0.4, 1
), nrow = 3, byrow = TRUE)

# Run the Monte Carlo simulation and print the results.
results <- mcs(num_sims, task_dists, cor_mat)
# print(results)
```

`print.pra_sigmoidal_fit` *Print method for Sigmoidal Model*

Description

Displays the summary of the fitted sigmoidal model in a readable format.

Usage

```
## S3 method for class 'pra_sigmoidal_fit'
print(x, ...)
```

Arguments

x	An object of class "pra_sigmoidal_fit" as returned by fit_sigmoidal.
...	Additional arguments (not used).

Value

No return value, called for side effects.

Examples

```
# Set up a data frame of time and completion percentage data
data <- data.frame(time = 1:10, completion = c(
  5, 15, 40, 60, 70, 75, 80, 85,
  90, 95
))
```

```
# Fit a logistic model to the data.
fit <- fit_sigmoidal(data, "time", "completion", "logistic")
# Print the model summary
print(fit)
```

---

print.smm *Print method for SMM results.*

Description

This function defines how to print the results of the Second Moment Method (SMM) analysis. It formats the output to display the total mean, variance, and standard deviation in a readable manner.

Usage

```
## S3 method for class 'smm'
print(x, ...)
```

Arguments

x	An object of class "smm" containing the SMM results.
...	Additional arguments (not used).

Value

None. The function prints the SMM results to the console.

Examples

```
mean <- c(10, 15, 20)
var <- c(4, 9, 16)
cor_mat <- matrix(c(
  1, 0.5, 0.3,
  0.5, 1, 0.4,
  0.3, 0.4, 1
), nrow = 3, byrow = TRUE)
result <- smm(mean, var, cor_mat)
print(result)

# Without correlation matrix (independent tasks)
result <- smm(mean, var)
print(result)
```

pv *Planned Value (PV).*

Description

Calculates the Planned Value (PV) of work completed based on the Budget at Completion (BAC) and the planned schedule.

Usage

```
pv(bac, schedule, time_period)
```

Arguments

bac	Budget at Completion (BAC) (total planned budget).
schedule	Vector of planned work completion (in terms of percentage) at each time period.
time_period	Current time period.

Value

The function returns the Planned Value (PV) of work completed.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

ev, ac, sv, cv, spi, cpi, eac

Examples

```
# Set the BAC, schedule, and current time period for a toy project.
bac <- 100000
schedule <- c(0.1, 0.2, 0.4, 0.7, 1.0)
time_period <- 3

# Calculate the PV and print the results.
pv <- pv(bac, schedule, time_period)
cat("Planned Value (PV):", pv, "\n")
```

retrieve_context *Retrieve Relevant Context for a Query*

Description

Searches the PRA knowledge base using combined vector similarity search (VSS) and BM25 full-text search to find the most relevant chunks for a user query.

Usage

```
retrieve_context(store, query, top_k = 3)
```

Arguments

store	A ragnar store object from `build_knowledge_base()`.
query	Character string. The user's question or query.
top_k	Integer. Number of chunks to retrieve (default 5).

Value

A character vector of relevant text chunks with source attribution, suitable for injecting into an LLM prompt as additional context.

Examples

```
## Not run:
store <- build_knowledge_base()
chunks <- retrieve_context(store, "What is earned value management?")
cat(chunks, sep = "\n---\n")

## End(Not run)
```

risk_post_prob *Posterior Risk Probability.*

Description

This function calculates the posterior probability of a risk event 'R' occurring based on observations of multiple root causes and their associated conditional probabilities.

Usage

```
risk_post_prob(
  cause_probs,
  risks_given_causes,
  risks_given_not_causes,
  observed_causes
)
```

Arguments

`cause_probs` A vector of prior probabilities for each root cause 'C_i'.

`risks_given_causes` A vector of conditional probabilities of the risk event 'R' given each cause 'C_i'.

`risks_given_not_causes` A vector of conditional probabilities of the risk event 'R' given not each cause 'C_i'.

`observed_causes` A vector of observed values for each cause 'C_i' (1 if observed, 0 if not observed, NA if unobserved).

Value

A numeric value for the posterior probability of the risk event given the observed causes.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
cause_probs <- c(0.3, 0.2)
risks_given_causes <- c(0.8, 0.6)
risks_given_not_causes <- c(0.2, 0.4)
observed_causes <- c(1, NA)
risk_post_prob <- risk_post_prob(
  cause_probs, risks_given_causes,
  risks_given_not_causes, observed_causes
)
print(risk_post_prob)
```

risk_prob *Bayesian Inference for Risk Probability.*

Description

This function calculates the overall probability of a risk event 'R' occurring based on the probabilities of multiple root causes and their associated conditional probabilities.

Usage

```
risk_prob(cause_probs, risks_given_causes, risks_given_not_causes)
```

Arguments

- `cause_probs` A vector of probabilities for each root cause 'C_i'.
- `risks_given_causes` A vector of conditional probabilities of the risk event 'R' given each cause 'C_i'.
- `risks_given_not_causes` A vector of conditional probabilities of the risk event 'R' given not each cause 'C_i'.

Value

The function returns a numeric value for the probability of risk event 'R'.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
cause_probs <- c(0.3, 0.2)
risks_given_causes <- c(0.8, 0.6)
risks_given_not_causes <- c(0.2, 0.4)
risk_prob_value <- risk_prob(cause_probs, risks_given_causes, risks_given_not_causes)
print(risk_prob_value)
```

`sensitivity` *Sensitivity Analysis.*

Description

This function performs sensitivity analysis on a project with multiple tasks, each having its own cost distribution. It calculates the sensitivity of the variance in total project cost with respect to the variance in each task's cost. It can also account for correlations between task costs if a correlation matrix is provided.

Usage

```
sensitivity(task_dists, cor_mat = NULL)
```

Arguments

`task_dists` A list of lists describing each task distribution. Each inner list should contain the type of distribution and its parameters. Supported distributions are "normal", "triangular", and "uniform".

`cor_mat` The correlation matrix for the tasks (Optional). If provided, it should be a square matrix with dimensions equal to the number of tasks. If not provided, tasks are assumed to be independent.

Value

The function returns a vector of sensitivity results with respect to each task. Each element in the vector corresponds to the sensitivity of the variance in total project cost with respect to the variance in the respective task's cost.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Set the task distributions for a toy project.
task_dists <- list(
  list(type = "normal", mean = 10, sd = 2), # Task A: Normal distribution
  list(type = "triangular", a = 5, b = 15, c = 10), # Task B: Triangular distribution
  list(type = "uniform", min = 8, max = 12) # Task C: Uniform distribution
)

# Set the correlation matrix between the tasks.
cor_mat <- matrix(c(
  1, 0.5, 0.3,
  0.5, 1, 0.4,
  0.3, 0.4, 1
```

```
), nrow = 3, byrow = TRUE)

# Calculate the sensitivity of each task and print the results
sensitivity_results <- sensitivity(task_dists, cor_mat)
print(sensitivity_results)

# Build a vertical barchart and display the results.
data <- data.frame(
  Tasks = c("A", "B", "C"),
  Sensitivity = sensitivity_results
)
barplot(
  height = data$Sensitivity, names = data$Tasks, col = "skyblue",
  horiz = TRUE, xlab = "Sensitivity", ylab = "Tasks"
)
title("Sensitivity Analysis of Project Tasks")

# Without correlation matrix
sensitivity_results_indep <- sensitivity(task_dists)
print(sensitivity_results_indep)

# Build a vertical barchart and display the results.
data_indep <- data.frame(
  Tasks = c("A", "B", "C"),
  Sensitivity = sensitivity_results_indep
)
barplot(
  height = data_indep$Sensitivity, names = data_indep$Tasks,
  col = "lightgreen",
  horiz = TRUE, xlab = "Sensitivity", ylab = "Tasks"
)
title("Sensitivity Analysis of Project Tasks (Independent)")
```

---

smm *Second Moment Method Analysis.*

Description

This function performs the Second Moment Method (SMM) analysis to estimate the total mean, variance, and standard deviation of a project based on individual task means, variances, and an optional correlation matrix.

Usage

```
smm(mean, var, cor_mat = NULL)
```

Arguments

mean	The mean vector.
var	The variance vector.
cor_mat	The correlation matrix (optional). If not provided, tasks are assumed to be independent.

Value

The function returns a list of the total mean, variance, and standard deviation for the project.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

Examples

```
# Set the mean vector, variance vector, and correlation matrix for a toy project.
mean <- c(10, 15, 20)
var <- c(4, 9, 16)
cor_mat <- matrix(c(
  1, 0.5, 0.3,
  0.5, 1, 0.4,
  0.3, 0.4, 1
), nrow = 3, byrow = TRUE)

# Use the Second Moment Method to estimate the results for the project.
result <- smm(mean, var, cor_mat)
print(result)

# Without correlation matrix (independent tasks)
result <- smm(mean, var)
print(result)

# When certain tasks are discrete and others are continuous, the SMM can still
# be applied as long as the variance values accurately reflect the variability of each task.

discrete_mean <- c(5, 10)
discrete_var <- c(0, 0)
continuous_mean <- c(15, 20)
continuous_var <- c(4, 5)
mean <- c(discrete_mean, continuous_mean)
var <- c(discrete_var, continuous_var)
cor_mat <- matrix(c(
  1, 0, 0.2, 0.3,
  0, 1, 0.1, 0.2,
  0.2, 0.1, 1, 0.4,
  0.3, 0.2, 0.4,
  1
), nrow = 4, byrow = TRUE)
result <- smm(mean, var, cor_mat)
```

```
print(result)
```

spi *Schedule Performance Index (SPI).*

Description

Calculates the Schedule Performance Index (SPI) of work completed based on the Earned Value (EV) and Planned Value (PV).

Usage

```
spi(ev, pv)
```

Arguments

ev	Earned Value.
pv	Planned Value.

Value

The function returns the Schedule Performance Index (SPI) of work completed.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

pv, ev, ac, sv, cv, cpi, eac

Examples

```
# Set the BAC, schedule, and current time period for an example project.
bac <- 100000
schedule <- c(0.1, 0.2, 0.4, 0.7, 1.0)
time_period <- 3

# Calculate the PV.
pv <- pv(bac, schedule, time_period)

# Set the actual % complete and calculate the EV.
actual_per_complete <- 0.35
ev <- ev(bac, actual_per_complete)

# Calculate the SPI and print the results.
spi <- spi(ev, pv)
cat("Schedule Performance Index (SPI):", spi, "\n")
```

sv — *Schedule Variance (SV).*

Description

Calculates the Schedule Variance (SV) of work completed based on the Earned Value (EV) and Planned Value (PV).

Usage

```
sv(ev, pv)
```

Arguments

ev	Earned Value.
pv	Planned Value.

Value

The function returns the Schedule Variance (SV) of work completed.

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

pv, ev, ac, cv, spi, cpi, eac

Examples

```
# Set the BAC, schedule, and current time period for an example project.
bac <- 100000
schedule <- c(0.1, 0.2, 0.4, 0.7, 1.0)
time_period <- 3

# Calculate the PV.
pv <- pv(bac, schedule, time_period)

# Set the actual % complete and calculate the EV.
actual_per_complete <- 0.35
ev <- ev(bac, actual_per_complete)

# Calculate the SV and print the results.
sv <- sv(ev, pv)
cat("Schedule Variance (SV):", sv, "\n")
```

tcpi *To-Complete Performance Index (TCPI).*

Description

Calculates the To-Complete Performance Index (TCPI), which indicates the cost performance required on remaining work to meet a target (BAC or EAC). TCPI > 1 means efficiency must improve; TCPI < 1 means efficiency can decrease.

Usage

```
tcpi(bac, ev, ac, target = "bac", eac = NULL)
```

Arguments

bac	Budget at Completion (BAC) (total planned budget).
ev	Earned Value.
ac	Actual Cost.
target	The target to calculate TCPI against. Either "bac" (default) to meet original budget, or "eac" to meet revised estimate. If "eac", the eac parameter must be provided.
eac	Estimate at Completion. Required when target = "eac".

Value

The function returns the To-Complete Performance Index (TCPI).

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

eac, etc, vac, cpi

Examples

```
bac <- 100000
ev <- 35000
ac <- 63000

# TCPI to complete within original budget
tcpi_bac <- tcpi(bac, ev, ac)
cat("TCPI (to meet BAC):", round(tcpi_bac, 2), "\n")

# TCPI to complete within revised estimate
```

```
eac <- 120482
tcpi_eac <- tcpi(bac, ev, ac, target = "eac", eac = eac)
cat("TCPI (to meet EAC):", round(tcpi_eac, 2), "\n")
```

---

vac *Variance at Completion (VAC).*

Description

Calculates the Variance at Completion (VAC), which is the difference between the budget and the expected final cost. Positive VAC indicates under budget, negative indicates over budget.

Usage

```
vac(bac, eac)
```

Arguments

bac	Budget at Completion (BAC) (total planned budget).
eac	Estimate at Completion.

Value

The function returns the Variance at Completion (VAC).

References

Damnjanovic, Ivan, and Kenneth Reinschmidt. Data analytics for engineering and construction project risk management. No. 172534. Cham, Switzerland: Springer, 2020.

See Also

eac, etc, tcpi

Examples

```
bac <- 100000
eac <- 120482 # From EAC calculation

vac <- vac(bac, eac)
cat("Variance at Completion (VAC):", round(vac, 2), "\n")
cat("Project is expected to be", abs(round(vac, 2)), ifelse(vac < 0, "over", "under"), "budget\n")
```

Index