Package: greenbook (via r-universe)

November 1, 2026

Title HM Treasury Green Book Cost-Benefit Analysis Primitives

Version 0.2.0

Description Implements cost-benefit analysis primitives from HM Treasury Green Book guidance (HM Treasury, 2026): the kinked Social Time Preference Rate (STPR) as revised in October 2026, with the earlier schedule retained for reproducibility, discount factors, net present value (NPV), equivalent annual cost, and real-terms rebasing using the GDP deflator. Designed for UK central government appraisal and evaluation. Bundled parameter tables carry vintage metadata for reproducibility.

Depends R (>= 4.1.0)

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Imports cli (>= 3.6.0), stats, utils

Suggests testthat (>= 3.0.0), knitr, rmarkdown, inflateR, openxlsx, officer, flextable

Config/testthat/edition 3

URL https://github.com/charlescoverdale/greenbook

BugReports https://github.com/charlescoverdale/greenbook/issues

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Charles Coverdale [aut, cre]

Maintainer Charles Coverdale <charlesfcoverdale@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-01 19:48:00 UTC

RemoteUrl https://github.com/cran/greenbook

RemoteRef HEAD

RemoteSha 6958111ee191449d839ef75ed67a7c28ad25717e

Contents

gb_apply_ob . . . 2
gb_appraise . . . 3
gb_carbon_npv . . . 5
gb_carbon_value . . . 6
gb_categories . . . 7
gb_compare . . . 8
gb_cost_per_unit . . . 9
gb_data_versions . . . 9
gb_deflator . . . 10
gb_discount . . . 11
gb_discount_factor . . . 12
gb_dist_weight . . . 13
gb_dist_weighted_npv . . . 14
gb_eanc . . . 15
gb_economic_case . . . 16
gb_headline . . . 17
gb_metb . . . 18
gb_npv . . . 19
gb_optimism_bias . . . 20
gb_place_based . . . 21
gb_progression . . . 22
gb_qaly . . . 23
gb_real . . . 24
gb_rebase . . . 25
gb_risk_register . . . 26
gb_schedule_table . . . 27
gb_sensitivity_ob . . . 28
gb_stpr . . . 29
gb_to_excel . . . 30
gb_to_latex . . . 31
gb_to_word . . . 32
gb_validate . . . 33
gb_vpf . . . 34
gb_wellby . . . 35

Index **37**

`gb_apply_ob` *Apply an optimism bias uplift to a cost stream*

Description

Applies an OB uplift, optionally with a mitigation factor that represents progress on risk identification and management: `cost_with_ob = cost_baseline * (1 + ob_pct * (1 - mitigation))`.

Usage

```
gb_apply_ob(values, ob_pct, mitigation = 0)
```

Arguments

values	Numeric vector of baseline cost values.
ob_pct	Numeric scalar. Optimism bias percentage as a decimal. Pass gb_optimism_bias(category) to use the published upper bound.
mitigation	Numeric scalar in [0, 1]. Fraction of the upper bound that has been mitigated through project definition and risk management. Default 0 (no mitigation; full upper bound applied).

Value

A numeric vector the same length as values, with the uplift applied.

References

HM Treasury (2003). Supplementary Green Book Guidance: Optimism Bias, Annex A2 on mitigation factors.

See Also

gb_optimism_bias(), gb_appraise().

Other optimism bias: gb_categories(), gb_optimism_bias()

Examples

```
costs <- c(100, 50, 50)
ob <- gb_optimism_bias("non_standard_buildings")
gb_apply_ob(costs, ob)
gb_apply_ob(costs, ob, mitigation = 0.5)
```

---

gb_appraise *Full Green Book appraisal in one call*

Description

Runs an end-to-end appraisal: optionally applies optimism bias to costs, optionally applies METB, computes NPV under the kinked STPR, and returns BCR alongside provenance.

Usage

```
gb_appraise(
  costs,
  benefits,
  years = NULL,
  schedule = "standard",
  ob = NULL,
  ob_dimension = "capex",
  ob_mitigation = 0,
  metb = FALSE,
  metb_rate = 0.2,
  base_year = NULL,
  vintage = "2026-10"
)
```

Arguments

costs	Numeric vector of cost values per period (real terms, base year fixed). Sign convention: enter as positive numbers; gb_appraise() handles the netting.
benefits	Numeric vector of benefit values per period. Same length as costs.
years	Optional integer vector of years. Defaults to 0:(length(costs) - 1).
schedule	One of "standard", "health", "catastrophic".
ob	Optional. Either a category name (character) or a numeric percentage. If supplied, optimism bias uplift is applied to costs.
ob_dimension	One of "capex" (default) or "duration". Only used when ob is a category name.
ob_mitigation	Numeric in [0, 1]. Mitigation fraction.
metb	Logical. If TRUE, applies METB uplift to costs.
metb_rate	Numeric. METB rate when metb = TRUE. Default 0.20 per Green Book 2022.
base_year	Optional integer base year.
vintage	Discounting vintage, "2026-10" (default) or "2003". Selects the STPR schedule (see gb_stpr()) and is stored on the returned object.

Value

A gb_appraisal object with extra fields bcr, pv_costs, pv_benefits, costs, benefits, optimism_bias, metb_applied.

References

HM Treasury (2026). The Green Book: Central Government Guidance on Appraisal and Evaluation.

See Also

gb_npv(), gb_optimism_bias(), gb_metb(), gb_dist_weighted_npv().

Other appraisal: gb_compare(), gb_economic_case(), gb_place_based(), gb_progression(), gb_risk_register(), gb_sensitivity_ob(), gb_validate()

Examples

```
costs <- c(100, 50, 50, 0, 0, 0, 0, 0, 0, 0)
benefits <- c(0, 0, 0, 30, 30, 30, 30, 30, 30, 30)
app <- gb_appraise(costs, benefits, ob = "non_standard_buildings",
                   ob_mitigation = 0.5, base_year = 2024)
app
```

---

gb_carbon_npv *Net present value of an emissions path*

Description

Multiplies an emissions vector (tCO2e per year) by the DESNZ carbon value at each year, then discounts under the kinked STPR. Returns a `gb_appraisal` object.

Usage

```
gb_carbon_npv(
  emissions,
  years,
  scenario = "central",
  schedule = "standard",
  base_year = NULL,
  sign = "cost",
  vintage = "2026-10"
)
```

Arguments

`emissions`	Numeric vector of emissions per year, in tCO2e (positive = emitted, negative = avoided / abated).
`years`	Integer vector of years matching `emissions`.
`scenario`	One of `"low"`, `"central"` (default), `"high"`.
`schedule`	One of `"standard"` (default), `"health"`, `"catastrophic"`.
`base_year`	Optional integer base year for monetary values (e.g. `2024`). If supplied, carbon values are rebased via `gb_deflator()`.
`sign`	Character. `"cost"` (default) treats positive `emissions` as a cost (negative cash-flow); `"benefit"` treats positive as avoided emissions (positive cashflow).
`vintage`	Discounting vintage, `"2026-10"` (default) or `"2003"`. See gb_stpr().

Value

A gb_appraisal object.

References

DESNZ (2023). Valuation of Energy Use and Greenhouse Gas Emissions for Appraisal (November 2023). HM Treasury (2026). The Green Book, chapter 8 (Valuation of social costs and social benefits).

See Also

gb_carbon_value(), gb_npv().

Other carbon: gb_carbon_value()

Examples

```
# 100 tCO2e emitted each year from 2024 to 2030
emissions <- rep(100, 7)
years <- 2024:2030
gb_carbon_npv(emissions, years, base_year = 2024)
```

---

gb_carbon_value *DESNZ carbon value for appraisal*

Description

Returns the DESNZ Carbon Value for Appraisal in GBP per tonne of CO2-equivalent at the published 2022 base-year prices, from the November 2023 valuation. Three scenarios: low, central, high.

Usage

```
gb_carbon_value(year, scenario = "central", base_year = NULL)
```

Arguments

year	Integer scalar or vector. Year of the emission. Must be within the bundled DESNZ range.
scenario	One of "low", "central" (default), "high".
base_year	Optional integer to rebase the value via gb_deflator(). If NULL, the published 2022 base year is used.

Details

DESNZ moved to a single consolidated series in November 2023, superseding the historical traded / non-traded split. Values are in 2022 prices. The November 2023 publication covers 2010 to 2100; the bundled subset is 2020 to 2050.

Value

Numeric vector of GBP per tCO2e values.

References

DESNZ (2023). Valuation of Energy Use and Greenhouse Gas Emissions for Appraisal (November 2023). Data tables 1-19, Table 3.

See Also

`gb_carbon_npv()`.

Other carbon: `gb_carbon_npv()`

Examples

```
gb_carbon_value(2024)
gb_carbon_value(2020:2030)
gb_carbon_value(2030, scenario = "high")
```

---

gb_categories *Available optimism bias categories*

Description

Returns the bundled OB category lookup as a data frame.

Usage

```
gb_categories()
```

Value

A data frame with columns `category`, `description`, `capex_upper`, `duration_upper`.

References

HM Treasury (2003). Supplementary Green Book Guidance: Optimism Bias.

See Also

`gb_optimism_bias()`.

Other optimism bias: `gb_apply_ob()`, `gb_optimism_bias()`

Examples

```
gb_categories()
```

gb_compare *Compare multiple appraisal options*

Description

Builds a side-by-side comparison of two or more gb_appraisal objects. Returns a gb_comparison with a ranked summary table and the preferred option under both NPV and BCR criteria.

Usage

```
gb_compare(..., by = "npv")
```

Arguments

...	gb_appraisal objects to compare. Pass as named arguments to control labels (e.g. do_minimum = app1).
by	Character. Ranking criterion: one of "npv" (default) or "bcr".

Details

Every Green Book economic case must compare at least Do Minimum against one or more Do Something options. gb_compare() standardises the comparison: NPV, BCR, EANC, PV costs, PV benefits, and rank under both NPV and BCR. The summary() method renders a one-page table suitable for a Five Case Model economic case.

Value

A gb_comparison object: a list with class c("gb_comparison", "list") and elements options, summary_table, preferred_npv, preferred_bcr, by.

References

HM Treasury (2026). The Green Book: Central Government Guidance on Appraisal and Evaluation, chapters 5 (Generating options and longlist appraisal) and 6 (Shortlist appraisal).

See Also

gb_appraise(), gb_economic_case().

Other appraisal: gb_appraise(), gb_economic_case(), gb_place_based(), gb_progression(), gb_risk_register(), gb_sensitivity_ob(), gb_validate()

Examples

```
do_minimum <- gb_appraise(c(50, 0, 0, 0, 0), c(0, 20, 20, 20, 20))
do_max <- gb_appraise(c(150, 0, 0, 0, 0), c(0, 50, 50, 50, 50))
gb_compare(do_minimum = do_minimum, do_max = do_max)
```

gb_cost_per_unit *Cost per unit delivered*

Description

Computes a cost-effectiveness ratio: total real cost (PV) per unit of monetised or non-monetised output. Standard cross-scheme comparator used by HM Treasury reviewers.

Usage

```
gb_cost_per_unit(appraisal, units_delivered, unit = "unit")
```

Arguments

appraisal	A gb_appraisal object.
units_delivered	Numeric scalar. The total quantity of the output being delivered (e.g. QALYs gained, tCO2e abated, WELLBYs added, jobs created).
unit	Character. Label for the unit (used in print).

Value

Numeric scalar: PV cost / units delivered, in GBP per unit.

See Also

gb_appraise().

Other reporting: gb_headline(), gb_to_excel(), gb_to_latex(), gb_to_word()

Examples

```
app <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 0, 0, 0, 0))
gb_cost_per_unit(app, units_delivered = 50, unit = "QALY")
```

gb_data_versions *Vintage of bundled parameter tables*

Description

Returns a data frame describing the source and last-updated date of every CSV bundled in inst/extdata/. Critical for reproducibility: every appraisal can record which vintage of Green Book parameters it used.

Usage

```
gb_data_versions()
```

Value

A data frame with columns `dataset`, `source`, `last_updated`, `notes`.

See Also

`gb_schedule_table()`.

Other lookups: `gb_schedule_table()`

Examples

```
gb_data_versions()
```

`gb_deflator` *GDP deflator factor between two years*

Description

Returns the multiplicative factor to convert a value expressed in year `from` prices to year `to` prices, using the bundled UK GDP deflator at market prices (calendar years). Multiply a nominal value from year `from` by this factor to express it in year `to`.

Usage

```
gb_deflator(from, to, source = "bundled")
```

Arguments

`from`	Integer year of the input value.
`to`	Integer year to express the value in.
`source`	Character. `"bundled"` uses the CSV shipped in `inst/extdata/`. Future versions will accept `"inflateR"` to use the live ONS series via the `inflateR` package.

Details

The bundled series is HM Treasury's October 2026 (Budget) release: ONS outturn to 2025 and OBR forecasts for 2026 to 2031. Factors that touch a forecast year are themselves forecasts and will be revised; the `type` column of the series (see `gb_data_versions()`) records which years are outturn.

Value

A numeric scalar: the deflator factor.

References

HM Treasury (2026). GDP deflators at market prices, and money GDP: October 2026 (Budget). https://www.gov.uk/government/collections/gdp-deflators-at-market-prices-and-money-gdp

See Also

gb_real(), gb_rebase().

Other deflator: gb_real(), gb_rebase()

Examples

```
gb_deflator(from = 2020, to = 2024)
gb_deflator(from = 2020, to = 2026) # 2026 is an OBR forecast year
```

---

gb_discount *Apply discount factors to a stream*

Description

Apply discount factors to a stream

Usage

```
gb_discount(
  values,
  years = seq_along(values) - 1L,
  schedule = "standard",
  vintage = "2026-10"
)
```

Arguments

values	Numeric vector of cashflow values (real terms, base year fixed).
years	Integer vector of years from the base year. Defaults to 0, 1, 2, ...
schedule	One of "standard", "health", "catastrophic".
vintage	Discounting vintage, "2026-10" (default) or "2003". See gb_stpr().

Value

A numeric vector of discounted (present-value) cashflows.

References

HM Treasury (2026). The Green Book, chapter 6 (Shortlist appraisal), section on discounting.

See Also

gb_discount_factor(), gb_npv().

Other discounting: gb_discount_factor(), gb_eanc(), gb_npv(), gb_stpr()

Examples

```
gb_discount(c(0, 100, 100, 100))
```

`gb_discount_factor` *Discount factor under the kinked STPR*

Description

Returns the present-value discount factor for each year, applying the kinked Social Time Preference Rate schedule annually.

Usage

```
gb_discount_factor(
  years,
  schedule = "standard",
  base_year = 0L,
  vintage = "2026-10"
)
```

Arguments

`years`	Integer vector of years from the base year. Must be `>= base_year`.
`schedule`	One of `"standard"`, `"health"`, `"catastrophic"`.
`base_year`	Integer base year offset. Default `0`.
`vintage`	Discounting vintage, `"2026-10"` (default) or `"2003"`. See `gb_stpr()`.

Details

The discount factor at year `t` is computed as the reciprocal of the cumulative product of `(1 + r_k)` for periods `k = 1, ..., t`, where `r_k` is the STPR for year `k`. This handles the kinked schedule correctly across band transitions (e.g. year 30 to year 31). Within a single band, the closed-form annuity factor `(1 - (1 + r)^(-n)) / r` agrees with `sum(gb_discount_factor(1:n))` to machine precision.

Value

A numeric vector of discount factors. 1 for `years == base_year`.

References

HM Treasury (2026). The Green Book, chapter 6 (Shortlist appraisal), section on discounting.

HM Treasury. Supplementary Green Book Guidance: Discounting. https://www.gov.uk/government/publications/green-book-supplementary-guidance-discounting

See Also

`gb_stpr()`, `gb_npv()`, `gb_eanc()`.

Other discounting: `gb_discount()`, `gb_eanc()`, `gb_npv()`, `gb_stpr()`

Examples

```
gb_discount_factor(0:5)
gb_discount_factor(c(0, 30, 31, 75, 76))
gb_discount_factor(c(0, 30, 31, 75, 76), vintage = "2003")
```

`gb_dist_weight` *Distributional weight for a recipient income*

Description

Returns the Green Book distributional weight applied to net benefits accruing to a recipient at income `income`, relative to a reference income (median by default), under iso-elastic utility:

Usage

```
gb_dist_weight(income, eta = 1.25, reference = "median", income_data = NULL)
```

Arguments

`income`	Numeric vector of recipient incomes (positive). Equivalised household disposable income is the conventional measure.
`eta`	Numeric scalar. Income elasticity of marginal utility of income. Default `1.25`.
`reference`	Either `"median"` (default; use median of `income` or `income_data` if supplied) or a numeric scalar (use that income as reference).
`income_data`	Optional numeric vector. Reference income distribution for computing the median. Use, e.g., the ONS household disposable income distribution. If `NULL` (default), the median of `income` is used.

Details

$$w_i = (y_{ref}/y_i)^{\eta}$$

Default `eta = 1.25`, the elasticity of marginal utility of consumption in the Green Book as updated on 28 October 2026, which adopted the estimate of the Green Book Discount Rate Review 2026. Earlier editions used 1.3; pass `eta = 1.3` to reproduce an appraisal made under them. Higher eta places greater weight on lower-income recipients.

Value

A numeric vector of weights, same length as `income`.

References

HM Treasury (2026). The Green Book, chapter 7 (Distributional analysis and place-based analysis), updated 28 October 2026.

Freeman, M. C. and Groom, B. (2026). Green Book Discount Rate Review: Technical Annexes. HM Treasury.

Acland, D. and Greenberg, D.H. (2024). The Elasticity of Marginal Utility of Income for Distributional Weighting and Social Discounting: A Meta-Analysis. Journal of Benefit-Cost Analysis.

See Also

gb_dist_weighted_npv().

Other distributional: gb_dist_weighted_npv()

Examples

```
# Weights across deciles of a stylised income distribution
income_deciles <- seq(10000, 100000, length.out = 10)
gb_dist_weight(income_deciles)
```

---

gb_dist_weighted_npv *Distributionally-weighted net present value*

Description

Applies recipient-income distributional weights to a cashflow before discounting under the STPR.

Usage

```
gb_dist_weighted_npv(
  cashflow,
  recipient_income,
  eta = 1.25,
  schedule = "standard",
  reference = "median",
  income_data = NULL,
  vintage = "2026-10",
  base_year = NULL
)
```

Arguments

cashflow	Numeric vector of net cashflows (per period).
recipient_income	Numeric vector. The income of the recipient (or representative recipient cell) in each period. Must have the same length as cashflow.
eta	Numeric scalar. Default 1.25.

schedule	One of "standard", "health", "catastrophic".
reference	Reference income strategy passed to gb_dist_weight().
income_data	Optional reference income distribution.
vintage	Discounting vintage, "2026-10" (default) or "2003". See gb_stpr().
base_year	Optional integer base year stored on the result.

Value

A gb_appraisal object with extra fields weights, eta, and unweighted_npv.

References

HM Treasury (2026). The Green Book, chapter 7 (Distributional analysis and place-based analysis).

See Also

gb_dist_weight(), gb_npv().

Other distributional: gb_dist_weight()

Examples

```
# 5-year benefit stream of GBP 30 going to a low-decile recipient
gb_dist_weighted_npv(
  cashflow = rep(30, 5),
  recipient_income = rep(15000, 5),
  income_data = seq(10000, 100000, length.out = 10)
)
```

---

gb_eanc *Equivalent annual net cost (or benefit)*

Description

Converts a present value to an annualised equivalent over a fixed horizon under the STPR. Used to compare options of different durations.

Usage

```
gb_eanc(npv, years, schedule = "standard", vintage = "2026-10")
```

Arguments

npv	Either a numeric scalar NPV, or a gb_appraisal object.
years	Project horizon in years. If npv is a gb_appraisal and years is missing, length(npv$cashflow) - 1L is used.
schedule	One of "standard", "health", "catastrophic".
vintage	Discounting vintage, "2026-10" (default) or "2003". Ignored when npv is a gb_appraisal, whose own schedule and vintage are used.

Value

A numeric scalar: the equivalent annual amount in real terms, base year aligned with the input.

References

HM Treasury. Supplementary Green Book Guidance: Discounting. Equivalent annual cost does not appear in the main text of the 2026 Green Book, which merged most annexes into the core chapters. https://www.gov.uk/government/publications/green-book-supplementary-guidance-discounting

See Also

gb_npv(), gb_appraise().

Other discounting: gb_discount(), gb_discount_factor(), gb_npv(), gb_stpr()

Examples

```
app <- gb_npv(c(-100, 30, 30, 30, 30, 30))
gb_eanc(app)
```

---

gb_economic_case *Render an appraisal as a Five Case Model economic case*

Description

Wraps a gb_appraisal with the structural sections HMT business case guidance expects in the Economic Case (the second of the five cases in the Five Case Model: Strategic, Economic, Commercial, Financial, Management).

Usage

```
gb_economic_case(
  appraisal,
  critical_success_factors = NULL,
  options_considered = NULL,
  non_monetised_impacts = NULL,
  recommendation = NULL,
  vfm_statement = NULL
)
```

Arguments

appraisal	A gb_appraisal (typically from gb_appraise() or gb_compare()).
critical_success_factors	Character vector of CSFs.
options_considered	Character vector of long-listed option names.

`non_monetised_impacts`
: Optional data frame with columns `impact`, `direction` (`"+"`/`"-"`), `materiality` (`"H"`/`"M"`/`"L"`), `notes`.

`recommendation`
: Optional character: the preferred option and rationale.

`vfm_statement`
: Optional character: the value-for-money judgment.

Details

The Five Case Model is HM Treasury's standard structure for business cases. The Economic Case is the part where Green Book appraisal sits: monetised costs and benefits, non-monetised impacts, switching values, sensitivity tests, value for money judgment, recommended option. `gb_economic_case` wraps the appraisal with the sections a reviewer expects to see.

Value

A `gb_economic_case` object.

References

HM Treasury (2018). Guide to Developing the Project Business Case (Five Case Model).

See Also

`gb_appraise()`, `gb_compare()`.

Other appraisal: `gb_appraise()`, `gb_compare()`, `gb_place_based()`, `gb_progression()`, `gb_risk_register()`, `gb_sensitivity_ob()`, `gb_validate()`

Examples

```
app <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30))
gb_economic_case(
  app,
  critical_success_factors = c("Strategic fit", "Value for money", "Achievability"),
  options_considered = c("Do nothing", "Do minimum", "Do maximum"),
  recommendation = "Do maximum: positive NPV and BCR > 1.5"
)
```

---

gb_headline *One-page headline summary of an appraisal*

Description

Returns the headline numbers a Green Book reviewer or steering committee expects: NPV, BCR, EANC, payback period, optimism bias applied, vintage. Suitable for a slide or executive summary.

Usage

```
gb_headline(appraisal)
```

Arguments

appraisal	A gb_appraisal object.

Value

A gb_headline object with elements npv, bcr, eanc, payback_year, optimism_bias, metb_applied, vintage, base_year, horizon.

See Also

gb_appraise(), gb_to_latex(), gb_to_excel().

Other reporting: gb_cost_per_unit(), gb_to_excel(), gb_to_latex(), gb_to_word()

Examples

```
app <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30))
gb_headline(app)
```

---

gb_metb *Apply Marginal Excess Tax Burden to public expenditure*

Description

Uplifts net public expenditure (or revenue raised) to reflect the welfare cost of distortionary taxation. Default rate is 20 percent, the value set out in the 2018 and 2022 Green Book editions. The historic value (2003) was 30 percent.

Usage

```
gb_metb(values, rate = 0.2, vintage = NULL)
```

Arguments

values	Numeric vector of expenditure values.
rate	Numeric scalar. METB rate as a decimal. Default 0.20.
vintage	Optional character. One of "2003", "2018", "2022", "2026". If supplied, overrides rate with the bundled value for that vintage. "2026" warns: see the section below.

Details

The METB captures the welfare loss arising from raising one extra GBP of revenue through the tax system, beyond the GBP itself. Estimates depend on the elasticity of taxable income, the marginal tax rate, and the distortionary margin. The Green Book reduced the headline value from 30 percent (2003) to 20 percent (2018) in light of post-2008 evidence.

Value

A numeric vector the same length as `values`, with the METB uplift applied.

Status under the 2026 Green Book

The 2026 Green Book does not specify an uplift, and states that "Practitioners should not generally include these costs in appraisal", on the grounds that most proposals are funded from pre-determined departmental budgets so the cost of raising the funds does not differ between the options being compared (chapter 6, "Costs of raising public funds"). The one exception it gives is the appraisal of private finance model options.

This function is therefore retained for historic and private-finance work rather than as a routine step. Applying it to a conventional 2026-basis appraisal will overstate costs. Requesting `vintage = "2026"` warns for that reason: the 20 percent figure is carried forward from 2022, not a value the 2026 edition sets.

References

HM Treasury (2026). The Green Book: Central Government Guidance on Appraisal and Evaluation, chapter 6 (Shortlist appraisal), section on the costs of raising public funds.

See Also

`gb_appraise()`.

Examples

```
gb_metb(c(100, 200))
gb_metb(c(100, 200), vintage = "2003")
```

---

gb_npv *Net present value*

Description

Computes the net present value of a cashflow stream under the Green Book STPR. Returns a `gb_appraisal` object carrying the NPV, the input cashflow, the discount factors, the schedule used, and methodology vintage.

Usage

```
gb_npv(
  cashflow,
  years = NULL,
  schedule = "standard",
  base_year = NULL,
  vintage = "2026-10"
)
```

Arguments

cashflow	Numeric vector of net cashflows in real terms (one value per year).
years	Integer vector of years matching cashflow. Defaults to 0, 1, ..., length(cashflow) - 1.
schedule	One of "standard", "health", "catastrophic".
base_year	Optional integer recording the price base year for the cashflow (e.g. 2024). Stored on the returned object.
vintage	Discounting vintage, "2026-10" (default) or "2003". Selects the STPR schedule (see gb_stpr()) and is stored on the returned object. Before version 0.2.0 this was a label only and defaulted to "2022".

Value

A gb_appraisal object: a list with class c("gb_appraisal", "list") and elements npv, cashflow, years, pv, schedule, base_year, vintage.

References

HM Treasury (2026). The Green Book, chapter 6 (Shortlist appraisal), section on discounting.

See Also

gb_appraise(), gb_eanc(), gb_dist_weighted_npv(), gb_carbon_npv().

Other discounting: gb_discount(), gb_discount_factor(), gb_eanc(), gb_stpr()

Examples

```
costs    <- c(100, 50, 50, 0, 0, 0, 0, 0, 0, 0)
benefits <- c(0, 0, 0, 30, 30, 30, 30, 30, 30, 30)
gb_npv(benefits - costs)
```

---

gb_optimism_bias *Optimism bias upper bound for a project category*

Description

Returns the indicative upper-bound optimism bias percentage from HM Treasury Supplementary Green Book Guidance: Optimism Bias (Mott MacDonald 2002, Annex A1). The upper bound is the starting value for ex-ante uplift; mitigation factors reduce it as project definition matures through SOC, OBC, and FBC stages.

Usage

```
gb_optimism_bias(category, dimension = "capex")
```

Arguments

category	Character. One of "standard_buildings", "non_standard_buildings", "standard_civil_engineer "non_standard_civil_engineering", "equipment_development", "outsourcing".
dimension	Character. "capex" (default) for capital expenditure uplift, or "duration" for works-duration uplift.

Value

Numeric scalar: the upper-bound percentage as a decimal (e.g. 0.51 for 51 percent).

References

HM Treasury (2003). Supplementary Green Book Guidance: Optimism Bias.

Mott MacDonald (2002). Review of Large Public Procurement in the UK. Report commissioned by HM Treasury.

See Also

gb_apply_ob(), gb_categories(), gb_appraise().

Other optimism bias: gb_apply_ob(), gb_categories()

Examples

```
gb_optimism_bias("non_standard_buildings")
gb_optimism_bias("standard_civil_engineering", dimension = "duration")
```

---

gb_place_based *Aggregate sub-projects into a place-based business case*

Description

Combines multiple gb_appraisal objects into a single portfolio appraisal. Introduced in the 2026 Green Book as a recognised business case structure for places (city deals, devolved settlements, regional packages) where individual projects are interdependent.

Usage

```
gb_place_based(..., place = NULL, synergy = 0)
```

Arguments

...	gb_appraisal objects representing sub-projects. Pass as named arguments to label each.
place	Character scalar. Place name (e.g. "Greater Manchester"). Optional.
synergy	Numeric scalar in [-0.5, 0.5]. Optional uplift applied to aggregate benefits to capture cross-project synergy or drag. Default 0 (additive aggregation).

Value

A gb_place_based object with elements projects, place, aggregate_npv, aggregate_bcr, per_project, synergy.

References

HM Treasury (2026). The Green Book, on place-based business cases.

See Also

gb_appraise(), gb_compare().

Other appraisal: gb_appraise(), gb_compare(), gb_economic_case(), gb_progression(), gb_risk_register(), gb_sensitivity_ob(), gb_validate()

Examples

```
transport <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30))
housing <- gb_appraise(c(50, 0, 0, 0, 0), c(0, 20, 20, 20, 20))
skills <- gb_appraise(c(20, 20, 0, 0, 0), c(0, 0, 15, 15, 15))
gb_place_based(transport = transport, housing = housing,
               skills = skills, place = "Example City")
```

---

gb_progression *Track an appraisal across SOC, OBC, and FBC stages*

Description

Builds a gb_progression showing how NPV, BCR, and key parameters evolve across the three Green Book business case stages: Strategic Outline Case (SOC), Outline Business Case (OBC), and Full Business Case (FBC). Useful for both authoring a multi-stage appraisal and reviewing how figures have moved between gateway approvals.

Usage

```
gb_progression(soc, obc = NULL, fbc = NULL)
```

Arguments

soc	Strategic Outline Case gb_appraisal.
obc	Outline Business Case gb_appraisal. Optional.
fbc	Full Business Case gb_appraisal. Optional.

Details

At each stage, optimism bias mitigation typically increases as project definition firms up. Cost estimates converge towards base-cost reality; benefit estimates may also shift as evidence accumulates. The evolution table makes the trajectory visible.

Value

A gb_progression object with elements stages, evolution (data frame), delta_npv, delta_costs.

References

HM Treasury (2018). Guide to Developing the Project Business Case (Green Book supplementary guidance).

See Also

gb_appraise().

Other appraisal: gb_appraise(), gb_compare(), gb_economic_case(), gb_place_based(), gb_risk_register(), gb_sensitivity_ob(), gb_validate()

Examples

```
soc <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30),
                   ob = "non_standard_buildings", ob_mitigation = 0)
obc <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30),
                   ob = "non_standard_buildings", ob_mitigation = 0.5)
fbc <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30),
                   ob = "non_standard_buildings", ob_mitigation = 0.9)
gb_progression(soc, obc, fbc)
```

---

gb_qaly *Value of a Quality-Adjusted Life Year*

Description

Returns the QALY value in GBP at the published base year. The DHSC supplementary Green Book guidance specifies GBP 70,000 per QALY in 2024 prices for cross-government appraisal. NICE Health Technology Assessment uses lower thresholds (GBP 20,000 to GBP 30,000 per QALY).

Usage

```
gb_qaly(qalys, scenario = "dhsc", base_year = NULL)
```

Arguments

qalys	Numeric scalar or vector. Quality-Adjusted Life Years gained or lost.
scenario	One of "dhsc" (default, GBP 70k 2024 prices, cross-government), "nice_lower" (GBP 20k), "nice_upper" (GBP 30k).
base_year	Optional integer base year. If NULL, the published base year for the chosen scenario is used.

Value

A numeric scalar or vector: monetised value in GBP.

References

DHSC Supplementary Green Book Guidance on health appraisal. NICE methods guides for technology appraisal.

See Also

`gb_wellby()`, `gb_vpf()`.

Other valuation: `gb_vpf()`, `gb_wellby()`

Examples

```
gb_qaly(1)
gb_qaly(1, scenario = "nice_upper")
```

---

gb_real *Convert nominal values to real*

Description

Converts nominal values at year-of-occurrence prices to real values at a chosen base year, using the bundled GDP deflator.

Usage

```
gb_real(nominal_values, year, base_year, source = "bundled")
```

Arguments

`nominal_values`	Numeric vector of nominal values.
`year`	Integer scalar or vector matching `nominal_values`, giving the year at which each value is expressed in nominal terms.
`base_year`	Integer scalar: the base year to convert to.
`source`	Character. `"bundled"` only in v0.1.0.

Value

A numeric vector of real values, in `base_year` prices.

References

HM Treasury (2026). The Green Book, chapter 6 (Shortlist appraisal), section on inflation.

See Also

`gb_deflator()`, `gb_rebase()`.

Other deflator: `gb_deflator()`, `gb_rebase()`

Examples

```
gb_real(nominal_values = c(100, 110, 120),
        year = 2020:2022,
        base_year = 2024)
```

`gb_rebase` *Rebase a real-terms series to a different base year*

Description

Multiplies values currently in `from`-year real prices by the deflator factor to express them in `to`-year real prices.

Usage

```
gb_rebase(values, from, to)
```

Arguments

`values`	Numeric vector of real-terms values.
`from`	Integer base year of the input series.
`to`	Integer target base year.

Value

A numeric vector of values in `to`-year real prices.

References

HM Treasury (2026). The Green Book, chapter 6 (Shortlist appraisal), section on inflation.

See Also

`gb_deflator()`, `gb_real()`.

Other deflator: `gb_deflator()`, `gb_real()`

Examples

```
gb_rebase(c(100, 200, 300), from = 2020, to = 2024)
```

gb_risk_register *Build a risk register with monetised exposure*

Description

Creates a gb_risk_register from a data frame of project risks. Computes expected loss per risk (probability times impact) and aggregate exposure. Optionally applied to a gb_appraisal to produce a risk-adjusted NPV.

Usage

```
gb_risk_register(risks, appraisal = NULL)
```

Arguments

risks	A data frame with columns id, description, probability (numeric in [0, 1]), impact_gbp (numeric). Optional columns: category, mitigation (character).
appraisal	Optional gb_appraisal. If supplied, returns a risk-adjusted NPV: appraisal$npv - sum(probability * impact).

Details

HM Treasury business case guidance requires a risk register with monetised exposure for the OBC and FBC stages. gb_risk_register standardises the structure: every risk has an id, a probability, a monetary impact, and an expected loss. Aggregation is by category if the column is present.

Value

A gb_risk_register object.

References

HM Treasury (2018). The Orange Book: Management of Risk - Principles and Concepts.

See Also

gb_appraise().

Other appraisal: gb_appraise(), gb_compare(), gb_economic_case(), gb_place_based(), gb_progression(), gb_sensitivity_ob(), gb_validate()

Examples

```
risks <- data.frame(
  id = c("R1", "R2", "R3"),
  description = c("Planning delay", "Cost overrun", "Lower demand"),
  category = c("schedule", "cost", "demand"),
  probability = c(0.30, 0.50, 0.20),
  impact_gbp = c(2e6, 5e6, 10e6)
```

```
)
gb_risk_register(risks)
```

`gb_schedule_table` *Tibble of the STPR schedule*

Description

Returns the bundled STPR kinked schedule as a data frame: one row per band, columns `year_from`, `year_to`, and (depending on `schedule`) either all three rate variants or just the requested one.

Usage

```
gb_schedule_table(schedule = NULL, vintage = "2026-10")
```

Arguments

`schedule`	Optional. One of `"standard"`, `"health"`, `"catastrophic"`. If `NULL` (default), all three columns are returned. The `"2026-10"` vintage has only `"standard"`; its other two columns are `NA`.
`vintage`	Discounting vintage, `"2026-10"` (default) or `"2003"`. See `gb_stpr()`.

Value

A data frame.

See Also

`gb_stpr()`, `gb_data_versions()`.

Other lookups: `gb_data_versions()`

Examples

```
gb_schedule_table()
gb_schedule_table("health", vintage = "2003")
```

gb_sensitivity_ob *Optimism bias sensitivity sweep*

Description

Recomputes an appraisal across a vector of optimism bias mitigation factors. Convenience wrapper for the standard HMT OB sensitivity test required at SOC, OBC, and FBC stages.

Usage

```
gb_sensitivity_ob(
  costs,
  benefits,
  ob,
  mitigations = c(0, 0.25, 0.5, 0.75, 1),
  ...
)
```

Arguments

costs	Numeric vector of cost values.
benefits	Numeric vector of benefit values.
ob	Either a category name or a numeric OB percentage.
mitigations	Numeric vector in [0, 1]. Mitigation fractions to test. Default c(0, 0.25, 0.5, 0.75, 1.0).
...	Passed to gb_appraise().

Value

A gb_sensitivity_ob object: a list with elements mitigations, npv, bcr, costs_pv.

See Also

gb_appraise(), gb_apply_ob().

Other appraisal: gb_appraise(), gb_compare(), gb_economic_case(), gb_place_based(), gb_progression(), gb_risk_register(), gb_validate()

Examples

```
costs <- c(100, 50, 50, 0, 0, 0, 0, 0, 0, 0)
benefits <- c(0, 0, 0, 30, 30, 30, 30, 30, 30, 30)
gb_sensitivity_ob(costs, benefits, ob = "non_standard_buildings")
```

gb_stpr *Social Time Preference Rate*

Description

Returns the HM Treasury Green Book Social Time Preference Rate (STPR) for a vector of years from a base year. The schedule is kinked: the rate steps down as the horizon lengthens. Two vintages are bundled.

Usage

```
gb_stpr(years, schedule = "standard", vintage = "2026-10")
```

Arguments

`years`	Integer vector of years from the base year (year 0 is the base year). Must be non-negative.
`schedule`	One of `"standard"` (default), `"health"` or `"catastrophic"`. The `"health"` (1.5 percent baseline, from DHSC supplementary guidance) and `"catastrophic"` (3.0 percent baseline, the standard schedule with the catastrophic-risk element of pure time preference excluded) schedules exist only in the `"2003"` vintage. The October 2026 guidance withdrew them: practitioners "should not use different discount rates for different types of proposals" (paragraph 3.2).
`vintage`	Discounting vintage. `"2026-10"` (default) is the schedule HM Treasury published on 28 October 2026 following the Green Book Discount Rate Review 2026: 3.00 percent for years 1 to 30, 2.50 percent for years 31 to 75 and 2.25 percent for years 76 to 125. `"2003"` is the schedule in force from the 2003 Green Book until October 2026: 3.5 percent declining to 1.0 percent. `"2026"` is accepted for `"2026-10"`, and `"2018"`, `"2020"`, `"2022"` and `"2026-02"` (the February 2026 edition, which kept the 3.5 percent schedule) are accepted for `"2003"`.

Details

The STPR is derived from the Ramsey formula: a risk-free rate (pure time preference, plus a wealth effect from expected per-capita consumption growth, plus an adjustment for rare disaster risk) and a systematic risk premium. In the 2026-10 vintage these are 0.50, 1.875, -0.50 and 1.00 percent. HM Treasury raises the resulting risk-free rate by 0.125 percentage points to 2.00 percent after triangulating against real gilt yields, which gives 3.00 percent. The kink reflects uncertainty about long-run growth and gilt yields.

The 2026-10 schedule stops at year 125. HM Treasury advises that discounting "is not an analytically meaningful technique" beyond that point (paragraph 3.10), so years above 125 are an error under that vintage rather than being silently extrapolated. Appraise later effects by other means, or use `vintage = "2003"` to reproduce an appraisal made under the earlier guidance.

Value

A numeric vector of discount rates (decimals, e.g. 0.03 for 3 percent), one per element of `years`.

References

HM Treasury (2026). The Green Book: Central Government Guidance on Appraisal and Evaluation, updated 28 October 2026.

HM Treasury (2026). Green Book supplementary guidance: discounting, updated 28 October 2026, paragraphs 3.2, 3.8 and 3.10. https://www.gov.uk/government/publications/green-book-supplementary-guidance

Freeman, M. C. and Groom, B. (2026). Green Book Discount Rate Review: Summary of Findings and Recommendations. HM Treasury.

See Also

`gb_discount_factor()`, `gb_discount()`, `gb_npv()`.

Other discounting: `gb_discount()`, `gb_discount_factor()`, `gb_eanc()`, `gb_npv()`

Examples

```
gb_stpr(0:5)
gb_stpr(c(10, 30, 31, 75, 76))
gb_stpr(c(10, 30, 31, 75, 76), vintage = "2003")
gb_stpr(c(10, 30, 31), schedule = "health", vintage = "2003")
```

---

gb_to_excel *Export an appraisal to Excel*

Description

Writes a multi-sheet workbook: summary, cashflow, present values, provenance.

Usage

```
gb_to_excel(appraisal, file)
```

Arguments

`appraisal`	A `gb_appraisal` object.
`file`	Output file path (must end in `.xlsx`).

Details

Requires the `openxlsx` package (in Suggests). Install with `install.packages("openxlsx")` if not present.

Value

Invisibly, the file path.

See Also

gb_to_latex(), gb_to_word().

Other reporting: gb_cost_per_unit(), gb_headline(), gb_to_latex(), gb_to_word()

Examples

```
if (requireNamespace("openxlsx", quietly = TRUE)) {
  app <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30))
  tmp <- tempfile(fileext = ".xlsx")
  gb_to_excel(app, tmp)
}
```

---

gb_to_latex *Render an appraisal as a LaTeX table*

Description

Render an appraisal as a LaTeX table

Usage

```
gb_to_latex(appraisal, caption = NULL, label = NULL)
```

Arguments

appraisal	A gb_appraisal object.
caption	Optional table caption.
label	Optional LaTeX label for cross-referencing.

Value

A character scalar containing a LaTeX tabular environment. Wrap in \\begin{table}...\\end{table} for a floating table.

See Also

gb_to_excel().

Other reporting: gb_cost_per_unit(), gb_headline(), gb_to_excel(), gb_to_word()

Examples

```
app <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30))
cat(gb_to_latex(app, caption = "Worked example"))
```

gb_to_word *Export an appraisal to Word*

Description

Writes a one-page Word document with the appraisal headline and cashflow table.

Usage

```
gb_to_word(appraisal, file)
```

Arguments

appraisal	A gb_appraisal object.
file	Output file path (must end in .docx).

Details

Requires the officer and flextable packages (both in Suggests).

Value

Invisibly, the file path.

See Also

gb_to_latex(), gb_to_excel().

Other reporting: gb_cost_per_unit(), gb_headline(), gb_to_excel(), gb_to_latex()

Examples

```
if (requireNamespace("officer", quietly = TRUE) &&
    requireNamespace("flextable", quietly = TRUE)) {
  app <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30))
  tmp <- tempfile(fileext = ".docx")
  gb_to_word(app, tmp)
}
```

gb_validate *Lint a Green Book appraisal for common errors*

Description

Inspects a gb_appraisal for sign-convention errors, base-year alignment, schedule plausibility, and other common authoring mistakes. Returns a structured report classifying each check as pass, warning, or error.

Usage

```
gb_validate(appraisal)
```

Arguments

appraisal	A gb_appraisal object.

Details

Checks performed:

- Cashflow length matches years length
- Schedule is one of the valid options
- Base year is plausible (1900 to 2100)
- Vintage matches a recognised Green Book edition
- If costs and benefits are present (from gb_appraise()), costs are non-negative and benefits are non-negative
- NPV is finite (no NaN / Inf)
- PV costs and PV benefits are consistent with NPV

Warnings flag suspicious patterns: missing base year, unusual optimism bias values, METB outside 0 to 50 percent, very long horizons (> 100 years).

Value

A gb_validation object with elements pass, errors, warnings, info, and n_checks.

See Also

gb_appraise().

Other appraisal: gb_appraise(), gb_compare(), gb_economic_case(), gb_place_based(), gb_progression(), gb_risk_register(), gb_sensitivity_ob()

Examples

```
app <- gb_appraise(c(100, 0, 0, 0, 0), c(0, 30, 30, 30, 30))
gb_validate(app)
```

gb_vpf *Value of Preventing a Fatality*

Description

Returns the DfT TAG-published Value of Preventing a Fatality (VPF) in real terms for a given year. The published 2024 value is GBP 2.153 million; the historical 2018 value (DfT TAG) is GBP 1.958 million. Years between bundled publication dates are filled by an annual real uplift of approximately 2 percent (proxy for real GDP per head growth).

Usage

```
gb_vpf(year = 2024, series = "central")
```

Arguments

`year`	Integer scalar. The year in which the value is expressed. Default `2024` (the most recent DfT-published value).
`series`	Character. `"central"` (default). Reserved for future expansion (DfT cancer / aversion multipliers).

Value

Numeric scalar: the VPF in GBP at year `year` prices.

References

Department for Transport. Transport Analysis Guidance (TAG) data book.

See Also

`gb_wellby()`, `gb_qaly()`.

Other valuation: `gb_qaly()`, `gb_wellby()`

Examples

```
gb_vpf()
gb_vpf(2018)
```

gb_wellby *Wellbeing valuation in GBP (WELLBY)*

Description

Monetises a life-satisfaction change as Well-being Adjusted Life Years (WELLBYs) per HMT Wellbeing Guidance for Appraisal (July 2021). One WELLBY equals a one-point change in life satisfaction on a 0 to 10 scale, for one person, for one year. The central published unit value is GBP 13,000 in 2019 prices, with a low-high sensitivity of GBP 10,000 to GBP 16,000.

Usage

```
gb_wellby(
  life_satisfaction_change,
  persons,
  years = 1,
  base_year = NULL,
  scenario = "central"
)
```

Arguments

life_satisfaction_change	Numeric scalar or vector. Change in life satisfaction on the 0 to 10 scale (signed; can be positive or negative).
persons	Number of people experiencing the change.
years	Duration in years. Default 1.
base_year	Optional integer base year to express the monetary value in. If NULL (default), the published 2019 price is returned. Otherwise the value is uplifted via gb_deflator() to base_year prices.
scenario	One of "low", "central" (default), "high".

Value

A numeric scalar or vector: the WELLBY value in GBP at the requested base year.

References

HM Treasury (2021). Wellbeing Guidance for Appraisal: Supplementary Green Book Guidance.

See Also

gb_vpf(), gb_qaly().

Other valuation: gb_qaly(), gb_vpf()

Examples

```
# 1-point lift in life satisfaction for 100 people for 5 years
gb_wellby(1, persons = 100, years = 5)
gb_wellby(1, persons = 100, years = 5, scenario = "low")
# Express in 2024 prices
gb_wellby(1, persons = 100, years = 5, base_year = 2024)
```

Index

∗ **adjustments**
gb_metb, 18
∗ **appraisal**
gb_appraise, 3
gb_compare, 8
gb_economic_case, 16
gb_place_based, 21
gb_progression, 22
gb_risk_register, 26
gb_sensitivity_ob, 28
gb_validate, 33
∗ **carbon**
gb_carbon_npv, 5
gb_carbon_value, 6
∗ **deflator**
gb_deflator, 10
gb_real, 24
gb_rebase, 25
∗ **discounting**
gb_discount, 11
gb_discount_factor, 12
gb_eanc, 15
gb_npv, 19
gb_stpr, 29
∗ **distributional**
gb_dist_weight, 13
gb_dist_weighted_npv, 14
∗ **lookups**
gb_data_versions, 9
gb_schedule_table, 27
∗ **optimism bias**
gb_apply_ob, 2
gb_categories, 7
gb_optimism_bias, 20
∗ **reporting**
gb_cost_per_unit, 9
gb_headline, 17
gb_to_excel, 30
gb_to_latex, 31
gb_to_word, 32
∗ **valuation**
gb_qaly, 23
gb_vpf, 34
gb_wellby, 35

gb_apply_ob, 2
gb_apply_ob(), *7*, *21*, *28*
gb_appraise, 3
gb_appraise(), *3*, *8*, *9*, *16–23*, *26*, *28*, *33*
gb_carbon_npv, 5
gb_carbon_npv(), *7*, *20*
gb_carbon_value, 6
gb_carbon_value(), *6*
gb_categories, 7
gb_categories(), *3*, *21*
gb_compare, 8
gb_compare(), *5*, *17*, *22, 23*, *26*, *28*, *33*
gb_cost_per_unit, 9
gb_cost_per_unit(), *18*, *31, 32*
gb_data_versions, 9
gb_data_versions(), *10*, *27*
gb_deflator, 10
gb_deflator(), *24, 25*
gb_discount, 11
gb_discount(), *12*, *16*, *20*, *30*
gb_discount_factor, 12
gb_discount_factor(), *11*, *16*, *20*, *30*
gb_dist_weight, 13
gb_dist_weight(), *15*
gb_dist_weighted_npv, 14
gb_dist_weighted_npv(), *5*, *14*, *20*
gb_eanc, 15
gb_eanc(), *11, 12*, *20*, *30*
gb_economic_case, 16
gb_economic_case(), *5*, *8*, *22, 23*, *26*, *28*, *33*
gb_headline, 17
gb_headline(), *9*, *31, 32*
gb_metb, 18
gb_metb(), *5*

gb_npv, 19
gb_npv(), *5, 6*, *11, 12*, *15, 16*, *30*
gb_optimism_bias, 20
gb_optimism_bias(), *3*, *5*, *7*
gb_place_based, 21
gb_place_based(), *5*, *8*, *17*, *23*, *26*, *28*, *33*
gb_progression, 22
gb_progression(), *5*, *8*, *17*, *22*, *26*, *28*, *33*
gb_qaly, 23
gb_qaly(), *34, 35*
gb_real, 24
gb_real(), *11*, *25*
gb_rebase, 25
gb_rebase(), *11*, *24*
gb_risk_register, 26
gb_risk_register(), *5*, *8*, *17*, *22, 23*, *28*, *33*
gb_schedule_table, 27
gb_schedule_table(), *10*
gb_sensitivity_ob, 28
gb_sensitivity_ob(), *5*, *8*, *17*, *22, 23*, *26*, *33*
gb_stpr, 29
gb_stpr(), *4, 5*, *11, 12*, *15, 16*, *20*, *27*
gb_to_excel, 30
gb_to_excel(), *9*, *18*, *31, 32*
gb_to_latex, 31
gb_to_latex(), *9*, *18*, *31, 32*
gb_to_word, 32
gb_to_word(), *9*, *18*, *31*
gb_validate, 33
gb_validate(), *5*, *8*, *17*, *22, 23*, *26*, *28*
gb_vpf, 34
gb_vpf(), *24*, *35*
gb_wellby, 35
gb_wellby(), *24*, *34*