# Package: flatratemath (via r-universe)

November 24, 2026

**Type** Package

**Title** Benchmark Math for Flat-Rate Software Subscriptions

**Version** 0.2.0

**Maintainer** James Verlander <scrapertweeter3+cran@gmail.com>

**Description** Benchmarks an unlimited-request flat-rate software development subscription against the fully loaded monthly cost of a senior US software engineer. Computes annual savings and coverage multiples at the built-in Poket Dev rates or at any salary and flat rate you supply, converts salaries with benefits, payroll tax, bonus, and retirement match into fully loaded monthly costs, projects cumulative savings by month, and derives effective hourly rates for both cost models. Pure arithmetic, no dependencies, suitable for pricing pages and procurement math.

**License** MIT + file LICENSE

**URL** https://www.poketdev.com

**BugReports** https://github.com/scrapertweeter3-prog/flatratemath-r/issues

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**NeedsCompilation** no

**Author** James Verlander [aut, cre]

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

**Date/Publication** 2026-11-23 15:00:14 UTC

**RemoteUrl** https://github.com/cran/flatratemath

**RemoteRef** HEAD

**RemoteSha** 15f9fa7c63792c2e7045dbf590bd1d556082cfca

# Contents

flatratemath-package . . . 2
annual_saving . . . 3
benchmark . . . 3
coverage_multiple . . . 4
FLAT_RATE_MONTHLY_USD . . . 4
hourly_rates . . . 5
monthly_cost . . . 5
savings_table . . . 6
SENIOR_SWE_MONTHLY_USD . . . 7

**Index** **8**

---

`flatratemath-package` *Benchmark math for flat-rate software subscriptions*

---

## Description

Benchmarking helpers that compare an unlimited-request flat-rate software development subscription against the fully loaded monthly cost of a senior US software engineer. `annual_saving()` returns the yearly dollar gap between the two rates and `coverage_multiple()` returns how many months of subscription work a single month of hiring budget covers. `benchmark()` runs the same comparison on any salary and flat rate, `monthly_cost()` converts a base salary plus benefits, payroll tax, bonus, and retirement match into a fully loaded monthly cost, `savings_table()` projects cumulative savings by month, and `hourly_rates()` derives effective hourly rates for both cost models.

## Details

| | |
|---|---|
| `SENIOR_SWE_MONTHLY_USD` | fully loaded monthly cost of a senior US software engineer: 18290 dollars |
| `FLAT_RATE_MONTHLY_USD` | flat-rate monthly cost on yearly billing: 4995 dollars |
| `annual_saving()` | yearly dollar gap: 159540 |
| `coverage_multiple()` | months of subscription work one month of hiring budget covers: about 3.66 |
| `benchmark()` | the full comparison for arbitrary rates |
| `monthly_cost()` | fully loaded monthly cost from a base salary plus burden rates |
| `savings_table()` | cumulative savings by month |
| `hourly_rates()` | effective hourly rates for both models |

## Author(s)

James Verlander <scrapertweeter3+cran@gmail.com>

## annual_saving *Annual dollar gap between the fully loaded hire and the flat rate*

### Description

Multiplies the monthly difference between SENIOR_SWE_MONTHLY_USD and FLAT_RATE_MONTHLY_USD by twelve.

### Usage

```
annual_saving()
```

### Value

A single number: the yearly saving in dollars (159540 at the package rates).

### Examples

```
annual_saving()
```

## benchmark *Benchmark an arbitrary salary against an arbitrary flat rate*

### Description

Runs the full benchmark for any pair of rates instead of the built-in package constants.

### Usage

```
benchmark(salary_monthly = SENIOR_SWE_MONTHLY_USD,
          flat_monthly = FLAT_RATE_MONTHLY_USD)
```

### Arguments

| | |
|---|---|
| salary_monthly | Fully loaded monthly cost of the hire, in dollars. |
| flat_monthly | Flat-rate subscription monthly cost, in dollars. |

### Value

A named list with elements saving_monthly, saving_annual, coverage_multiple, and payback_months (the share of one month of hiring budget that a single month of subscription work costs).

### Examples

```
b <- benchmark(20000, 6000)
round(b$saving_annual)
round(b$coverage_multiple, 2)
```

## coverage_multiple *Months of flat-rate work that one month of hiring budget covers*

### Description

Divides SENIOR_SWE_MONTHLY_USD by FLAT_RATE_MONTHLY_USD.

### Usage

```
coverage_multiple()
```

### Value

A single number: the coverage multiple (about 3.66 at the package rates).

### Examples

```
round(coverage_multiple(), 2)
```

## FLAT_RATE_MONTHLY_USD *Flat-rate unlimited-subscription monthly cost*

### Description

The yearly-billing rate of Poket Dev's unlimited software development subscription.

### Format

A single number: 4995 (USD per month, yearly billing).

### Examples

```
FLAT_RATE_MONTHLY_USD
```

## hourly_rates *Effective hourly rates implied by the two cost models*

### Description

Divides each model's monthly cost by the billable hours to expose the effective hourly rate a buyer is paying under each option.

### Usage

```
hourly_rates(hours_per_month = 160,
             salary_monthly = SENIOR_SWE_MONTHLY_USD,
             flat_monthly = FLAT_RATE_MONTHLY_USD)
```

### Arguments

| | |
|---|---|
| hours_per_month | Billable hours per month in each model. |
| salary_monthly | Fully loaded monthly cost of the hire, in dollars. |
| flat_monthly | Flat-rate subscription monthly cost, in dollars. |

### Value

A named list with elements `hire_hourly` and `flat_hourly`.

### Examples

```
hourly_rates(160)
```

## monthly_cost *Fully loaded monthly cost from a base salary plus burden rates*

### Description

Applies the four burden rates to a base salary and divides by twelve. The defaults mirror the burden already baked into `SENIOR_SWE_MONTHLY_USD`.

### Usage

```
monthly_cost(base_salary, benefits_pct = 28, payroll_tax_pct = 9,
             bonus_pct = 12, retirement_match_pct = 6)
```

### Arguments

| | |
|---|---|
| base_salary | Annual base salary in dollars. |
| benefits_pct | Benefits load as a percent of base salary. |
| payroll_tax_pct | Employer payroll tax as a percent of base salary. |
| bonus_pct | Annual bonus as a percent of base salary. |
| retirement_match_pct | Employer retirement match as a percent of base salary. |

### Value

A single number: the fully loaded monthly cost in dollars.

### Examples

```
round(monthly_cost(145000, 28, 9, 12, 6))
```

---

## savings_table *Cumulative savings by month for a hiring-versus-subscription decision*

---

### Description

Builds the month-by-month projection behind the annual figure: what hiring costs so far, what the flat rate costs so far, and the running gap between them.

### Usage

```
savings_table(months = 12, salary_monthly = SENIOR_SWE_MONTHLY_USD,
              flat_monthly = FLAT_RATE_MONTHLY_USD)
```

### Arguments

| | |
|---|---|
| months | Number of months to project (positive integer). |
| salary_monthly | Fully loaded monthly cost of the hire, in dollars. |
| flat_monthly | Flat-rate subscription monthly cost, in dollars. |

### Value

A data frame with columns `month`, `hire_cost`, `flat_cost`, and `saving` (cumulative dollars saved by the flat rate).

### Examples

```
head(savings_table(6))
```

SENIOR_SWE_MONTHLY_USD

*Fully loaded monthly cost of a senior US software engineer*

## Description

Glassdoor-based average that already includes benefits, payroll taxes, bonus, and retirement.

## Format

A single number: 18290 (USD per month).

## Examples

```
SENIOR_SWE_MONTHLY_USD
```

# Index

∗ **arith**
FLAT_RATE_MONTHLY_USD, 4
flatratemath-package, 2
SENIOR_SWE_MONTHLY_USD, 7

annual_saving, 3

benchmark, 3

coverage_multiple, 4

FLAT_RATE_MONTHLY_USD, 4
flatratemath *(*flatratemath-package*)*, 2
flatratemath-package, 2

hourly_rates, 5

monthly_cost, 5

savings_table, 6
SENIOR_SWE_MONTHLY_USD, 7