Package ‘CompModels’

January 20, 2025

Title Pseudo Computer Models for Optimization

Version 0.3.0

Description A suite of computer model test functions that can be used to test and evaluate algorithms for Bayesian (also known as sequential) optimization. Some of the functions have known functional forms, however, most are intended to serve as black-box functions where evaluation requires running computer code that reveals little about the functional forms of the objective and/or constraints. The primary goal of the package is to provide users (especially those who do not have access to real computer models) a source of reproducible and shareable examples that can be used for benchmarking algorithms. The package is a living repository, and so more functions will be added over time. For function suggestions, please do contact the author of the package.

License GPL-2

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation yes

Suggests R.rsp, laGP

VignetteBuilder R.rsp

Author Tony Pourmohamad [aut, cre]

Maintainer Tony Pourmohamad <tpourmohamad@gmail.com>

Repository CRAN

Date/Publication 2022-10-20 14:07:58 UTC

Contents

`bbox1` *A black-box computer model*

Description

A black-box computer model that evaluates the objective and constraint functions for a given input value.

Usage

```
bbox1(x1, x2)
```

Arguments

x1	A scalar value between -1.5 and 2.5, inclusive
x2	A scalar value between -3 and 3, inclusive

Value

The evaluation of running the black-box computer model at input (x1, x2).

- obj: A scalar objective function value
- con: A vector of constraint function values

Note

A solution is feasible only if of all of the constraint functions values are less than or equal to 0.

Examples

```
### Running the black-box computer model at x1 = 1, x2 = 2
### obj = -1.5 and con = (-3.5,-0.25)
### (x1,x2) = (1,2) is a feasible solution

bbox1(1,2)
```

bbox2 *A black-box computer model*

Description

A black-box computer model that evaluates the objective function for a given input value.

Usage

```
bbox2(x1, x2)
```

Arguments

x1	A scalar value between -3 and 3, inclusive
x2	A scalar value between -2 and 2, inclusive

Value

The evaluation of running the black-box computer model at input (x1, x2).

- obj: A scalar objective function value

Examples

```
### Running the black-box computer model at x1 = 1, x2 = 1
### obj = 3.23333

bbox2(1,1)
```

bbox3 *A black-box computer model*

Description

A black-box computer model that evaluates the objective function for a given input value.

Usage

```
bbox3(x1, x2)
```

Arguments

x1	A scalar value between -2 and 2, inclusive
x2	A scalar value between -2 and 2, inclusive

Value

The evaluation of running the black-box computer model at input (x1, x2).

- obj: A scalar objective function value

Examples

```
### Running the black-box computer model at x1 = 1, x2 = 1
### obj = 1876

bbox3(1,1)
```

---

bbox4 *A black-box computer model*

Description

A black-box computer model that evaluates the objective and constraint functions for a given input value.

Usage

```
bbox4(x1, x2)
```

Arguments

x1	A scalar value between -10 and 0, inclusive
x2	A scalar value between -6.5 and 0, inclusive

Value

The evaluation of running the black-box computer model at input (x1, x2).

- obj: A scalar objective function value
- con: A vector of constraint function values

Note

A solution is feasible only if all of the constraint function values are less than or equal to 0.

Examples

```
### Running the black-box computer model at x1 = -1, x2 = -1
### obj = 15.00539 and con = 7
### (x1,x2) = (-1,-1) is not a feasible solution

bbox4(-1,-1)
```

bbox5 *A black-box computer model*

Description

A black-box computer model that evaluates the objective function for a given input value.

Usage

```
bbox5(x1, x2, x3)
```

Arguments

`x1`	A scalar value between -pi and pi, inclusive
`x2`	A scalar value between -pi and pi, inclusive
`x3`	A scalar value between -pi and pi, inclusive

Value

The evaluation of running the black-box computer model at input (x1, x2, x3).

- obj: A scalar objective function value

Examples

```
### Running the black-box computer model at x1 = 1, x2 = 1, x3 =1
### obj = 5.840058

bbox5(1,1,1)
```

bbox6 *A black-box computer model*

Description

A black-box computer model that evaluates the objective and constraint functions for a given input value.

Usage

```
bbox6(x1)
```

Arguments

`x1`	A scalar value between 0 and 10, inclusive

Value

The evaluation of running the black-box computer model at input x1.

- obj: A scalar objective function value
- con: A vector of constraint function values

Note

A solution is feasible only if the all of the constraint function values are less than or equal to 0.

Examples

```
### Running the black-box computer model at x1 = 1
### obj = 0.926574 and con = (0.4259819,0.6472136)
### x1 = 1 is not a feasible solution

bbox6(1)
```

`bbox7` *A black-box computer model*

Description

A black-box computer model that evaluates the objective and constraint functions for a given input value.

Usage

```
bbox7(x1, x2, x3, x4, x5, x6, x7, x8)
```

Arguments

`x1`	A scalar value between 0 and 1, inclusive
`x2`	A scalar value between 0 and 1, inclusive
`x3`	A scalar value between 0 and 1, inclusive
`x4`	A scalar value between 0 and 1, inclusive
`x5`	A scalar value between 0 and 1, inclusive
`x6`	A scalar value between 0 and 1, inclusive
`x7`	A scalar value between 0 and 1, inclusive
`x8`	A scalar value between 0 and 1, inclusive

Value

The evaluation of running the black-box computer model at input (x1, x2, x3, x4, x5, x6, x7, x8).

- obj: A scalar objective function value
- con: A vector of constraint function values

Note

A solution is feasible only if all of the constraint function values are less than or equal to 0.

Examples

```
### Running the black-box computer model at x1 = 0, x2 = 0, x3 = 0, x4 = 0,
###                                         x5 = 0, x6 = 0, x7 = 0, x8 = 0
### obj = 1 and con = (0.2,-0.8)
### (x1,x2,x3,x4,x5,x6,x7,x8) = (0,0,0,0,0,0,0,0) is not a feasible solution

bbox7(0,0,0,0,0,0,0,0)
```

gram *The Gramacy et. al 2016 test function*

Description

The computer model test function introduced in Gramacy et. al 2016.

Usage

```
gram(x1, x2)
```

Arguments

x1	A scalar value between 0 and 1, inclusive
x2	A scalar value between 0 and 1, inclusive

Value

The evaluation of running the Gramacy et. al 2016 test function at input (x1, x2).

- obj: A scalar objective function value
- con: A vector of constraint function values

Note

A solution is feasible only if of all of the constraint functions values are less than or equal to 0.

References

Gramacy, R.B., Gray, G.A., Digabel, S.L., Lee, H.K.H., Ranjan, P., Wells, G., and Wild, S.M. (2016). Modeling an augmented Lagrangian for blackbox constrained optimization. Technometrics, 58(1):1-11.

Examples

```
### Running the function at x1 = 0.2 and x2 = 0.2
### obj = 0.4 and con = (1.29,-1.42)
### (x1,x2) = (0.2,0.2) is not a feasible solution

gram(0.2,0.2)
```

`mtp` *The modified Townsend function*

Description

This function evaluates the objective and constraint functions for the modified Townsend function.

Usage

```
mtp(x1, x2)
```

Arguments

`x1`	A scalar value between -2.25 and 2.5, inclusive
`x2`	A scalar value between -2.5 and 1.75, inclusive

Value

The evaluation of running the modified Townsend function at input pair (x1, x2).

- obj: A scalar objective function value
- con: A scalar constraint function value

Note

A solution is feasible when the constraint function is less than or equal to 0.

Examples

```
### Running the modified Townsend function at x1 = 1 and x2 = 0.8
### obj = 0.9073035 and con = 2.012166
### (x1,x2) = (1,1) is not a feasible solution

mtp(1,1)
```

pressure *The pressure vessel computer model*

Description

The pressure vessel computer model is designed to minimize the total cost of constructing a pressure vessel considering the cost of material, forming, and welding. The four inputs to the computer model are the thickness of the shell (x1), the thickness of the head (x2), the inner radius (x3), and the length of the cylindrical section of the vessel (x4) not including the head. Note, the thicknesses of the variables are integer multiples of 0.0625 inches. The cost of the pressure vessel is subject to four constraints.

Usage

```
pressure(x1, x2, x3, x4)
```

Arguments

x1	A scalar value between 0 and 99, inclusive, controlling the thickness of the shell.
x2	A scalar value between 0 and 99, inclusive, controlling the thickness of the head.
x3	A scalar value between 0 and 200, inclusive, controlling the inner radius.
x4	A scalar value between 0 and 200, inclusive, controlling the length of the cylindrical section of the vessel (x4) not including the head.

Value

The evaluation of running the tension spring computer model at input (x1, x2, x3).

- obj: A scalar objective function value
- con: A vector of constraint function values

Note

A solution is feasible only if of all of the constraint functions values are less than or equal to 0.

Examples

```
### Running the function at x1 = 10, x2 = 4, x3 = 21, x4 = 24.
### obj = 55536.1 and con = (-10.4053, -20.77104, 1219107.94901, -216)
### (x1,x2,x3,x4) = (10,4,21,24) is not a feasible solution

pressure(10,4,21,24)
```

sprinkler *The spinkler computer model*

Description

The sprinkler computer model is a multiobjective optimization problem. The objectives are to minimize the water consumption associated with using a garden sprinkler, while also maximizing the speed and range of the garden sprinkler. The eight inputs to the computer model are the vertical (x1) and tangential (x2) nozzle angle, the nozzle profile (x3), the diameter of the sprinkler head (x4), the dynamic (x5) and static (x6) friction moment, the entrance pressure (x7), and the diameter flow line (x8).

Usage

```
sprinkler(x1, x2, x3, x4, x5, x6, x7, x8)
```

Arguments

`x1` A scalar value between 0 and 90, inclusive, controlling the vertical nozzle angle.

`x2` A scalar value between 0 and 90, inclusive, controlling the tangential nozzle angle.

`x3` A scalar value between 2e-6 and 4e-6, inclusive, controlling the nozzle profile.

`x4` A scalar value between 0.1 and 0.2, inclusive, controlling the diameter of the sprinkler head.

`x5` A scalar value between 0.01 and 0.02, inclusive, controlling the dynamic friction moment.

`x6` A scalar value between 0.01 and 0.02, inclusive, controlling the static friction moment.

`x7` A scalar value between 1 and 2, inclusive, controlling the entrance pressure.

`x8` A scalar value between 5 and 10, inclusive, controlling the diameter flow line.

Value

The evaluation of running the sprinkler computer model at input (x1, x2, x3, x4, x5, x6, x7, x8).

- obj: A vector of objective function values for consumption, speed, and range (in that order)

Note

The order of the outputs of the sprinkler computer model is consumption (minimize), speed (maximize), and range (maximize).

References

Bebber, D. v., Hochkirchen, T., Siebertz, K. (2010). Statistische Versuchsplanung: Design of Experiments (DoE). Germany: Springer Berlin Heidelberg.

Examples

```
### Running the function at x1 = 33, x2 = 18, x3 = 2e-6, x4 = 0.18,
###                         x5 = 0.015, x6 = 0.0199, x7 = 1.54, x8 = 7.5.
### obj = (4.218397, 5.321142, 3.124046)

sprinkler(33,18,2e-6,0.18,0.015,0.0199,1.54,7.5)
```

tension *The tension spring computer model*

Description

The tension spring computer model is designed to minimize the weight of a tension spring. The three inputs to the computer model are the wire diameter (x1), mean coil diameter (x2), and the number of active coils (x3). The tension spring weight is subject to four constraints on the shear stress, surge frequency, and deflection.

Usage

```
tension(x1, x2, x3)
```

Arguments

`x1` A scalar value between 0.05 and 2, inclusive, controlling the wire diameter.

`x2` A scalar value between 0.25 and 1.3, inclusive, controlling the mean coil diameter.

`x3` A scalar value between 2 and 15, inclusive, controlling the number of active coils.

Value

The evaluation of running the tension spring computer model at input (x1, x2, x3).

- obj: A scalar objective function value
- con: A vector of constraint function values

Note

A solution is feasible only if of all of the constraint functions values are less than or equal to 0.

Examples

```
### Running the function at x1 = 1, x2 = 1, x3 = 3.
### obj = 5 and con = (1,45.82,-1,0.33)
### (x1,x2,x3) = (1,1,3) is not a feasible solution

tension(1,1,3)
```

Index