smbo Documentation

Release 0.1

Drew Blount

March 19, 2015

CONTENTS

built as part of Drew Blount's Senior Mathematics Thesis at Reed College

CHAPTER

ONE

INDICES AND TABLES

- genindex
- modindex
- search

1.1 Installation

Install smbo by running:

```
install smbo
```

1.2 Source

https://github.com/drewblount/2014-2015/tree/master/thesis/code/smbo

CHAPTER

TWO

DOCUMENTATION FOR THE CODE

2.1 doctest – including docstrings

This is something I want to say that is not in the docstring.

class smbo.smb_optimizer.**smb_optimizer**(*domain*, *objective_func*, *modeller*, *init_sampler=None*)

An object that, given an input domain, objective function, and modelling strategy, seeks to efficiently find the global optimum of the objective by the generation of sequential models.

Parameters

- **domain** (*list*) – a list whose i^{th} element is the $(lower\ bound,\ upper\ bound)$ pair describing the domain of interest in the i^{th} input dimension. The length of this list defines the dimension of input space, denoted k. This smb_optimizer then optimizes the k-rectangle defined by the domain arg.
- **objective_func** (*function*) – a function (or any object with a suitable __apply__ method) that maps k-vectors to floats. The goal of an smb_optimizer is to minimize this function over the domain defined above.
- **modeller** (*function*) – a function (or any object with a suitable __apply__ method) that maps a tuple (X, Y), where :math‘X‘ is a list of sample points (each a k-vector from the domain), and Y their evaluated objective values; to a tuple of functions $(\hat{y},\ \hat{\sigma}^2)$. Each output function maps points in the input domain to real numbers. $\hat{y}$ represents the model's best estimate of the objective function, and $\hat{\sigma}^2$ is the estimated error of the prediction $\hat{y}$.
- **init_sampler** (*function*) – a function which will select initial sample points, informing the zero-generation model. If left unspecified, is by default set to a $2k + 2$-sample latin hypercube over the domain, created with smbo.latin_hypercube.

Returns An object with attributes duplicating domain, objective_func, and modeller. Further, the class method initialize() is called, which picks initial sample points, evaluates the objective function at those points, and generates an inital predictive model with modeller. This object is then ready to iteratively improve its predictions and seek the global minimum of objective_func; the object now has fields self.X, self.Y, self.y_hat, and self.err_hat to describe the sample points, corresponding objective values, predictor function, and certainty function respectively.

choose_sample(*randomize=False*)

Parameters **randomize** (*bool*) – if true, the next sample point is chosen randomly with probability weighted by expected improvement; otherwise, returns the point in the input domain with the highest expected improvement.

Chooses the next sample point

find_min()

Returns {x: xcoord, y: ycoord}, simply the (x,y) sample point with the lowest y value (the current best-minimum)

Return type dict

class smbo.models.**dace**(*X*, *Y*)

A little hefty to be simply a function, this class (which behaves as a function because of its __apply__ method) fits a DACE model to provided sample points

Parameters

- **X** (*list*) – a list of input vectors
- **Y** (*list*) – a list of observed objective values

Returns (pred_y,pred_err): two functions, each k-to-1, where k is the dimension of the input space, representing the dace predictor surface and predicted error at each point in input space

Return type tuple

conc_likelihood(*new_P=None*, *new_Q=None*)

Parameters

- **new_P** (*list*) – an n-vector resetting the p parameter of the DACE model
- **new_Q** (*list*) – an n-vector resetting the q or :math:‘ heta‘ parameter of the DACE model

Returns the likelihood of the current DACE params P and Q, given the data X and Y

Return type float

corr(*x1*, *x2*)

Parameters

- **x1** (*list*) – a coordinate in the domain
- **x2** (*list*) – a coordinate in the domain

Returns the correlation between estimation errors at x1 and x2

Return type float

corr_vector(*x_new*)

Parameters **x_new** – a k-vector from the domain

Returns a vector whose i^{th} element is the parameterized correlation between x_new and the :math‘i^{th}‘ sample point

Return type list

dist(*x1*, *x2*)

Parameters

- **x1** (*list*) – a coordinate in the domain
- **x2** (*list*) – a coordinate in the domain

Returns the parameterized distance between x1 and x2

Return type float

max_likelihood(*bounds=None*, *verbose=False*)

Parameters **bounds** (*list*) – the $PimesQ$ domain over which likelihood is being maximized, defaults to $(1, 2)imes(0, \infty)$

Returns res: an object describing the P and :math‘Q‘ values that optimize likelihood

Return type optimization_result

The evaluation of this function also resets self.P and self.Q to the values indicated by res, i.e. it sets P and Q to maximize the likelihood of the DACE model, thereby fitting the model to the data.

pred_err(*x_new*)

Parameters **x_new** (*list*) – a k-vector from the domain

Returns the predicted function value at x_new

Return type float

This is computed using the so-called best linear unbiased predictor, Jones Eq. 7

predict(*x_new*)

Parameters **x_new** (*list*) – a k-vector from the domain

Returns the predicted function value at x_new

Return type float

This is computed using the so-called best linear unbiased predictor, Jones Eq. 7

reset()

Resets all lazyprops

smbo.samplers.**latin_hypercube**(*m*, *k*, *bounds=None*, *rand_sampler=<built-in method random of Random object>*)

Parameters

- **m** (*int*) – the number of desired sample points
- **k** (*int*) – the dimension of input space
- **bounds** (*list*) – the k min-max tuples describing the function domain as a k-rectangle. Defaults to the unit k-cube.
- **rand_sampler** (*function*) – used to choose actual sample coordinates once the latin hypercube selects a sample's particular hyper(sub)rectangle in the input domain.

Returns An m-list of k-vectors, representing an m-point latin hypercube sample of the k-dimensional input domain.

Return type list

smbo.lazyprop.**lazyprop**(*fn*)

Parameters **fn** (*function*) – a function, whose only argument is self, whose output shouldn't be computed more than once for a given X,Y pair.

Returns

_lazyprop: a function that calls fn the first time it is called, then remembers that output and returns this remembered value after subsequent calls

Return type function

smbo.lazyprop.**reset_lps**(*self*)

Resets all lazyprops, so that the evaluated function vals are forgotten and must be recomputed from raw data

2.1.1 License

The project is licensed under the MIT license.

l

m

s