

apsis

Automated Hyperparameter Optimization Using Bayesian Optimization

Frederik Diehl

Andreas Jauch

March 10, 2015

AGENDA

Problem Description

Bayesian Optimization

apsis and its Architecture

Project Organisation

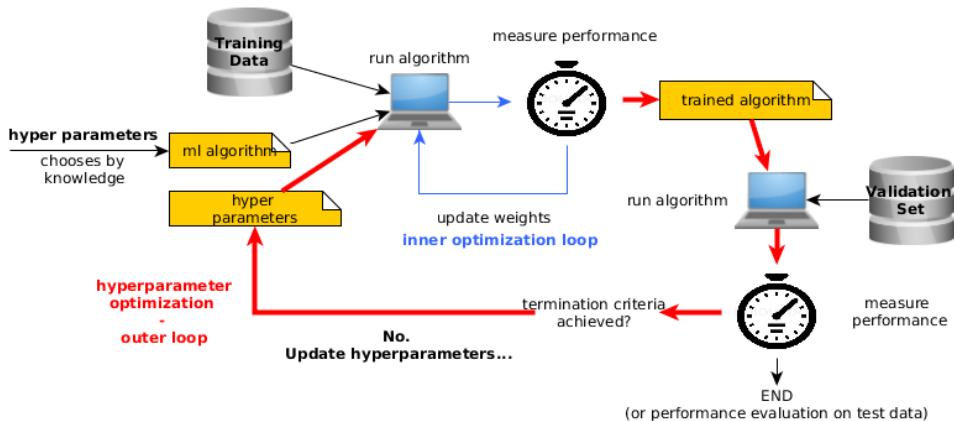
Performance Evaluation

MOTIVATION

Why Hyperparameter Optimization and why automating it?

- ▶ hyperparameter tuning often leads to huge performance gain
- ▶ "more of an art than a science"
- ▶ reproducibility of published results
- ▶ automatic methods might be better than humans
- ▶ provide ml algorithms to non-expert users

ML PROCESS OVERVIEW



FORMAL PROBLEM DESCRIPTION

λ : hyperparameter vector $\lambda = (\lambda^{(1)}, \dots, \lambda^{(n)})$

$L(X, f)$: loss function evaluated for model f and dataset x

$A_\lambda(X)$: learning algorithm with hyperparameter vector λ
learning on dataset X

X_{train} : training data, X_{valid} : validation data, X_{test} : test data

$\Psi(\lambda)$: hyperparameter response function/surface

Hyperparameter Optimization Problem

$$\hat{\lambda} \approx \underset{\lambda \in \Lambda}{\operatorname{argmin}} \underbrace{\left(\operatorname{mean}_{X_i \in X_{\text{valid}}} (L(x_i, A_\lambda(X_{\text{train}}))) \right)}_{\Psi(\lambda)} \quad (1)$$

$$= \underset{\lambda \in \Lambda}{\operatorname{argmin}} (\Psi(\lambda)) \quad (2)$$

KEY PROBLEM PROPERTIES

- ▶ unknown, probably non-convex response surface Ψ
- ▶ no derivative-based optimization possible
- ▶ every evaluation of Ψ is expensive
- ▶ evaluation time of Ψ depends on individual value of λ
- ▶ low effective dimensionality of Ψ
- ▶ which dimensions are important is dataset dependent
- ▶ tree-structured configuration space¹

¹not addressed in *apsis* yet

STATE OF THE ART

- ▶ optimization still manual in many projects
- ▶ grid search most common method
 - ▶ often the only provided method by many ml frameworks
- ▶ random search
- ▶ Bayesian Optimization
 - ▶ code of Jasper Snoek et. al. Harvard/Toronto
 - ▶ whetlab - bay opt in the cloud

BAYESIAN OPTIMIZATION

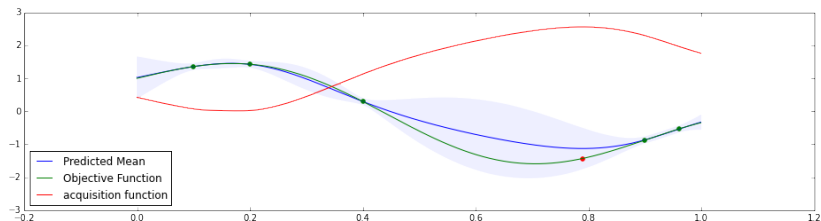
- ▶ approximate $\Psi(\lambda)$ by a *surrogate* function $M(\lambda) = y$
- ▶ surrogate function cheaper to evaluate than Ψ
- ▶ interpret model to find minimization candidates for Ψ
- ▶ evaluate Ψ for promising candidates

BAYESIAN OPTIMIZATION FUNDAMENTALS

We need two design choices

- ▶ Surrogate Modelling Function - Gaussian Processes
 - ▶ universal approximation
 - ▶ very flexible and have many useful properties
 - ▶ closed under sampling
- ▶ Acquisition Function
 - ▶ Probability of Improvement
 - ▶ Expected Improvement

ACQUISITION FUNCTION u



- measures the expected utility of evaluating the objective function at a point λ_{next}
- exploitation vs. exploration trade-off

OPTIMIZATION - SUCCESSIVELY UPDATING THE GP

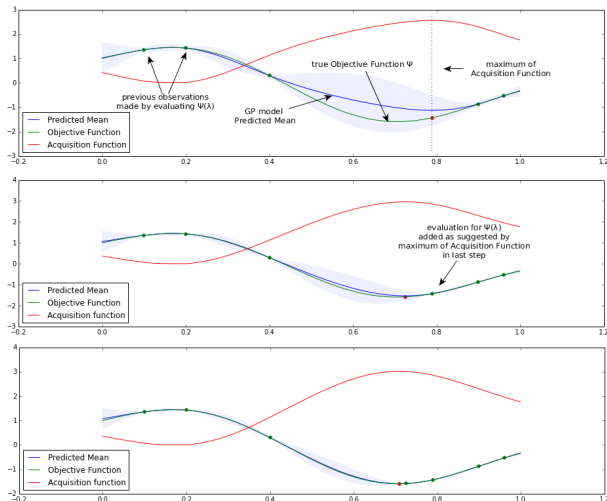
1. find

$$\max_{\lambda} (u(\lambda)) = \lambda_{\text{next}}$$

max of acquisition

2. Evaluate M at λ_{next}

3. Update the GP



FITTING THE GP TO THE PROBLEM

by tuning the covariance!

- Squared Exponential Kernel

$$K_{SE}(\lambda, \lambda') = \exp \left(-\frac{1}{2l^2} \cdot \sum_{1..dim(D)} (\lambda_d - \lambda'_d)^2 \right)$$

- use Automatic Relevance Determination (ARD)

$$K_{SE}(\lambda, \lambda') = \theta_0 \cdot \exp \left(-\frac{1}{2} \cdot \sum_{1..dim(D)} \left(\frac{1}{\theta_d^2} (\lambda_d - \lambda'_d)^2 \right) \right)$$

with ARD vector θ

$$\theta = \underbrace{(\theta_0)}_{\text{bias}}, \underbrace{(\theta_1, \dots, \theta_d)}_{\text{dimension weights}}$$

HOW DO ACQUISITION FUNCTIONS LOOK LIKE?

Expected Improvement (EI)

$$u_{\text{EI}}(\lambda) = \int_{-\infty}^{\infty} \max(\Psi(\lambda^*) - y, 0) \cdot p_M(y|\lambda) dy$$

- ▶ closed form solution for GPs available

$$u_{\text{EI}}(\lambda|M_t) = \sigma(\lambda) \cdot \left(\frac{f(\lambda^*) - \mu(\lambda)}{\sigma(\lambda)} \cdot \Phi(\lambda) + \phi(\lambda) \right)$$

- ▶ gradient analytically derived in *apsis* for more effective optimization

THE *apsis* TOOLKIT

Automated Hyperparameter Optimization Framework for

- ▶ random search
- ▶ Bayesian Optimization

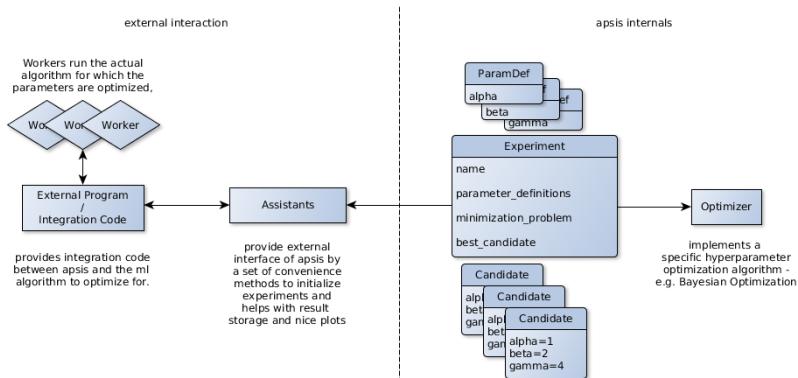
as an open source framework featuring

- ▶ flexible architecture, ready to be extended for more optimizers
- ▶ ready for use with scikit-learn and theano
- ▶ implemented in Python

PROJECT OBJECTIVES

- ▶ open source implementation of state of the art research in Bayesian Optimization
- ▶ extendible project to encourage collaboration with other researchers
- ▶ easy integration with existing machine learning frameworks
- ▶ multi core support

apis ARCHITECTURE OVERVIEW



apsis CORE MODEL COMPONENTS

Parameter Definitions

- ▶ define the meta information for each hyperparameter

Candidates

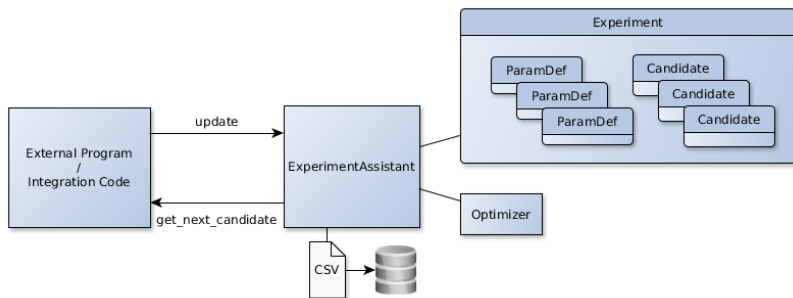
- ▶ represent a specific hyperparameter vector and its value
- ▶ holds function value if available

Experiments

- ▶ represent an optimization object
- ▶ keeps track of finished and unfinished Candidates

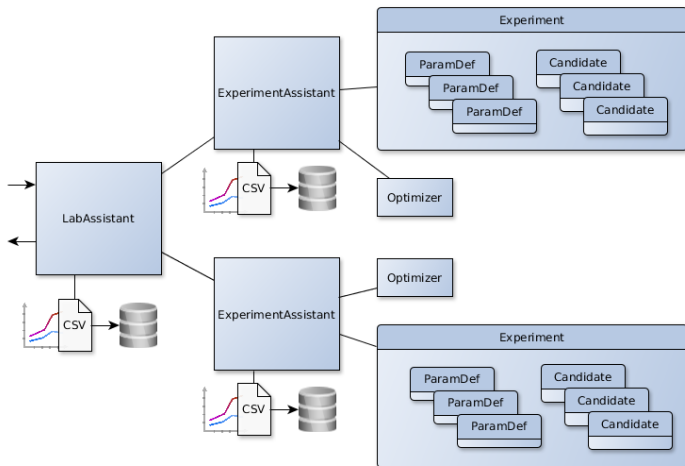
USING *apsis* - EXPERIMENT ASSISTANTS

- ▶ single experiment interaction interface
- ▶ provides plots and result bookkeeping



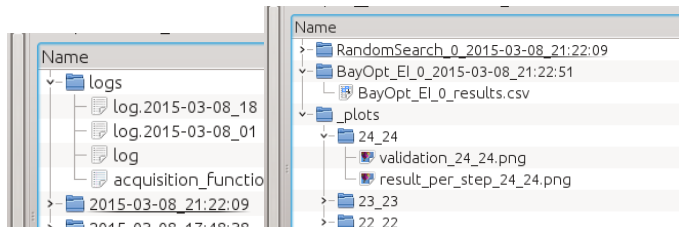
USING *apsis* - LAB ASSISTANTS

- ▶ multiple experiments to compare different optimization techniques
- ▶ cross validation



EXTENSIVE EXPERIMENT TRACKING

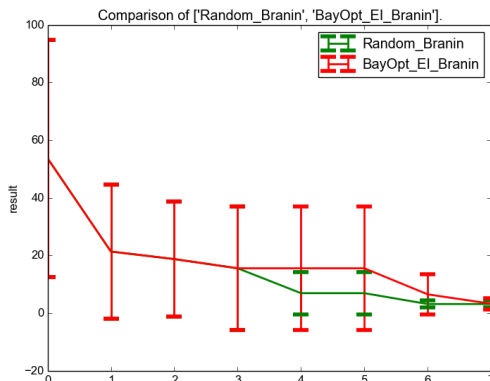
- ▶ automated plot writing
- ▶ automated results writing
- ▶ write out information at every step



	A	B	C	D	E	F
1	step	x	y	cost	result	best_result
2	1	-1.8149133398	12.486639612	None	17.7445201863	2.3755334224
3	2	9.2332830588	0.9757738948	None	2.3755334224	2.3755334224
4	3	8.4224102564	14.0924841235	None	156.939061018	2.3755334224
5	4	7.2319214268	0.0828317569	None	16.9493267082	2.3755334224
6	5	3.4191579635	11.4117757293	None	88.0623767792	2.3755334224
7	6	7.1218056935	13.9454647851	None	178.418789129	2.3755334224
8	7	0.0642275711	1.79798051	None	36.395028905	2.3755334224
9	8	2.3417914042	2.1734230814	None	3.9615520468	2.3755334224

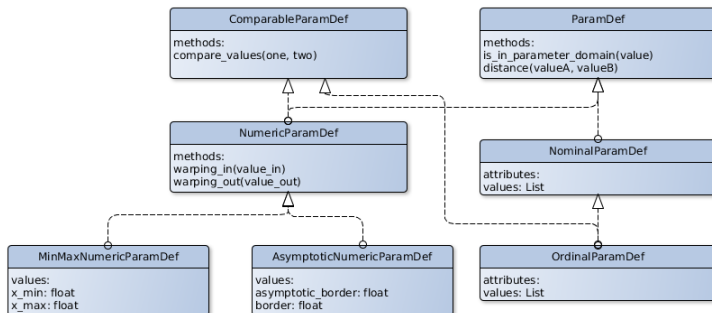
AUTOMATED PLOTTING

- ▶ plot function evaluations and best results
- ▶ plot confidence bars when using cross validation
- ▶ write out plots at every step



PARAMETER REPRESENTATION IN *apis*

- ▶ different representation by parameter type
- ▶ various nominal and numeric types



NUMERIC PARAMETERS IN *apsis*

Warping Mechanism

- ▶ parameters are warped into $[0, 1]$ interval
- ▶ optimization core can assume a uniform and equal distribution in $[0, 1]$ space
- ▶ warping can be user defined

Provided Warpings for

- ▶ normalization of arbitrary intervals $[a, b]$ into $[0, 1]$ space.
- ▶ asymptotic parameters, e.g. learning rate asymptotic at 0

NOMINAL PARAMETERS IN *apsis*

- ▶ generally supported in *apsis*
 - ▶ no support in Bayesian Optimization
 - ▶ GP kernels based on distance metrics between parameters
-
- ▶ interesting topic for further research
 - ▶ no publications on this topic so far
 - ▶ whetlab pretends to deal well with them - but doesn't say how

EXPECTED IMPROVEMENT OPTIMIZATION

- ▶ gradient analytically derived²
- ▶ 1000 Steps Random Search for Initialization
- ▶ Several iterative optimization methods integrated
 - ▶ L-BFGS-B Bounded Low Memory Quasi Newton Method
 - ▶ BFGS Quasi Newton Method
 - ▶ Nelder-Mead
 - ▶ Inexact Newton with Conjugate Gradient Solver
 - ▶ ...

²See our paper for derivation.

DEALING WITH GP HYPERPARAMETERS

- ▶ the GP surrogate model introduces new hyperparameter
 - ▶ not subject of optimization \Rightarrow hyper-hyperparameters
- ▶ optimization by maximum likelihood method
- ▶ integrating over these parameters in the acquisition function using Hybrid Monte Carlo sampling

apsis PROJECT SET UP

- ▶ Open-Source project from the beginning
- ▶ MIT-License
- ▶ active issue tracking
- ▶ PEP-8 code styling convention
- ▶ Fully automated sphinx documentation build on every commit
- ▶ 90% test coverage
- ▶ clear commit messages

apsis GITHUB REPOSITORY

The screenshot shows the GitHub repository page for 'apsis'. At the top, it displays repository statistics: 396 commits, 5 branches, 0 releases, and 3 contributors. Below this is a blue header bar with the repository name 'apsis' and a dropdown menu showing 'branch: master'. A green 'clone' button is on the left, and a menu icon is on the right. The main content area shows a commit message: 'added mac os installation tutorial doc to documentation.' by user 'andi1400' 15 hours ago. Below the commit message is a table of files changed in the commit:

File	Change	Time
code/apsis	added mac os installation tutorial doc to documentation.	15 hours ago
diagrams	updated param defs picture.	2 days ago
documentation	added mac os installation tutorial doc to documentation.	15 hours ago
.gitignore	updated gitignore.	6 days ago
License.txt	Changed readme and added license file.	4 months ago
README.md	Now hosting the docs at http://apsis.readthedocs.org . Updated readme.	19 days ago

Below the file list is a section for the 'README.md' file, which contains the word 'apsis' in a large, bold font.

Check out <http://github.com/FrederikDiehl/apsis>!

ISSUE TRACKING AND DISCUSSION

Optimization of EI always gets stuck in local extrema #31

 Closed

andi1400 opened this issue on 31 Oct 2014 · 7 comments

Edit

New Issue



andi1400 commented on 31 Oct 2014

Collaborator

At the moment we optimize EI with `scipy.optimize.minimize`. Tried several optimizer methods, all of them get stuck in local extrema.

Problem seems to be well known:

- the [bergstra group](http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf) reports on it <http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf> (page 3)
- <http://www.icml2010.org/papers/297.pdf> report about this, too
- the old `spearint` package also seems to use grid search here if I understand their code right

Proposed optimization methods there

- grid search (probably not feasible)
- evolutionary algorithm

Other ideas

- random search, how many points?
- continue using SLSQP, L-BFGS-B and implement a random restart for them. How often?



bayerj commented on 31 Oct 2014

Collaborator

Climin has a recent evolutionary algorithm. <https://github.com/BRML/climin/blob/master/climin/nas.py>

I say continue with grid search first, solve the problem later.



FrederikDiehl commented on 31 Oct 2014

Owner

I agree with that. GridSearch (or random search) should suffice for now. After all, the main problem will be inverting the matrix, not evaluating it.

 FrederikDiehl referenced this issue from a commit on 4 Nov 2014

 #31. Fixed the implementation of grid search for the acquisition func. 

c184763

Labels

Issues

Pull requests

Labels

Milestones

Filters

is:issue

 Clear current search query, filters, and sorts

Asst

No

Not


You

3 pr

1 t

 4 Open  98 Closed



Author Labels Milestone

 **Parameter Defs: include a preconstructed one for asymptotic functions**  **enhancement**
#103 opened 3 days ago by FrederikDiehl

 **ValidationLabHelper - Possibility to initialize experiments with the same random run.**  **enhancement**
#102 opened 6 days ago by andi1400

 **EI Optiization ignores valid parameter range**  **bug**
#101 opened 8 days ago by andi1400

 **Noise generating function**  **enhancement**
#100 opened 9 days ago by FrederikDiehl 

 **Logging: init only works for some directories**  **bug**
#99 opened 10 days ago by FrederikDiehl

 **ValidationLabHelper**  **enhancement**
#98 opened 12 days ago by FrederikDiehl

 **ComparableParameterDef rename to ComparableParamDef for consistency**
#97 opened 14 days ago by andi1400

 **Experiment does not test for the correctness of candidates**  **bug**
#96 opened 15 days ago by FrederikDiehl

 **Complicated Tests to Demo**
#95 opened 18 days ago by FrederikDiehl

 **Write Unittests**
#94 opened 18 days ago by FrederikDiehl 

UNIT TESTS

```

[fred@fred-W540 aphis]$ nosetests --with-coverage --cover-package aphis
/usr/lib/python2.7/site-packages/GPy/util/linalg.py:48: UserWarning: warning: caught this exception:'module' object has no attribute '_dotblas'
  warnings.warn("warning: caught this exception:" + str(e))
.....
-----
Name                               Stmts  Miss  Cover   Missing
-----
apsis                               0      0   100%
apsis.assistants                   0      0   100%
apsis.assistants.experiment_assistant 114      3    97% 199, 283, 354
apsis.assistants.lab_assistant     230     30    87% 268, 424-444, 490-491, 535, 576, 597-601, 622-625, 750-762
apsis.demos                        1      0   100%
apsis.demos.demo_MNIST             56     19    66% 40, 43-47, 50, 70, 85, 93-96, 99, 110, 118, 120-133, 136
apsis.demos.demo_MNIST_MCMC        19      4    79% 34, 36-38, 41
apsis.models                       0      0   100%
apsis.models.candidate             51      0   100%
apsis.models.experiment            126      3    98% 357-359, 405
apsis.models.parameter_definition   171     22    87% 34, 41-42, 81, 337, 346-350, 448-453, 474, 476, 483-488
apsis.optimizers                   0      0   100%
apsis.optimizers.bayesian           1      0   100%
apsis.optimizers.bayesian.acquisition_functions 201     15    93% 59, 74-75, 131-137, 295, 393-395, 460-461, 484, 494, 509, 542
apsis.optimizers.bayesian_optimization 87      1    99% 223
apsis.optimizers.optimizer          21     11    48% 26, 45, 63-66, 82-87
apsis.optimizers.random_search     28      0   100%
apsis.utilities                    0      0   100%
apsis.utilities.file_utils          4      0   100%
apsis.utilities.import_utils        9      3    67% 24-27
apsis.utilities.logging_utils       28      0   100%
apsis.utilities.optimizer_utils     12      2    83% 36-37
apsis.utilities.plot_utils          69     12    83% 62, 65, 104-107, 177, 224, 246, 257, 262, 266
apsis.utilities.randomization       12      2    83% 20, 23
-----
TOTAL                             1240    127    90%
-----
Ran 34 tests in 70.357s
OK

```

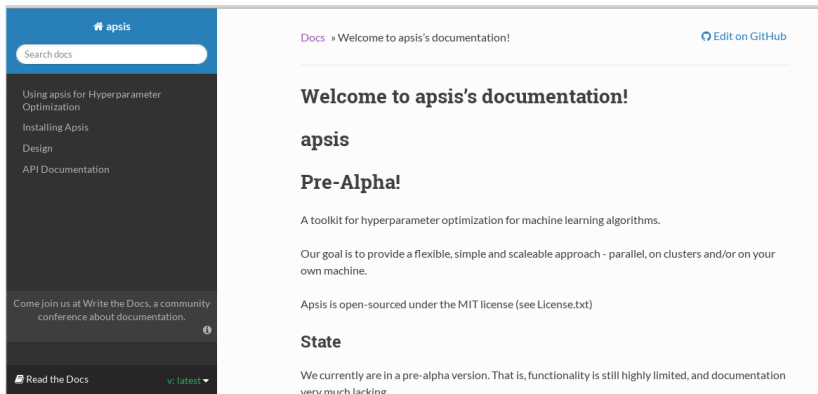
- ▶ 90% overall test coverage
- ▶ 100% in most core components

GOOD CODE DOCUMENTATION

```
http://cloc.sourceforge.net v 1.62  T=0.16 s (224.5 files/s, 33127.1 lines/s)
-----
Language             files            blank          comment          code
-----
Python                36              865             2039             2408
-----
SUM:                  36              865             2039             2408
-----
```

- ▶ almost 50:50 ratio of code vs. doc
- ▶ documented according to sphinx standard

FULLY AUTOMATED DOCUMENTATION BUILD

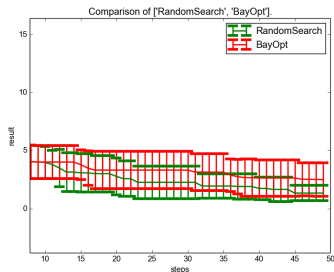


The screenshot displays the Apsis documentation website. On the left is a dark sidebar with a blue header containing the 'apsis' logo and a search bar. The sidebar lists navigation items: 'Using aphis for Hyperparameter Optimization', 'Installing Aphis', 'Design', and 'API Documentation'. At the bottom of the sidebar, it says 'Come join us at Write the Docs, a community conference about documentation.' and 'Read the Docs' with a version selector set to 'v: latest'. The main content area has a light gray background. It features a breadcrumb 'Docs » Welcome to aphis's documentation!' and a link to 'Edit on GitHub'. The main heading is 'Welcome to aphis's documentation!' followed by the 'apsis' logo and 'Pre-Alpha!'. The text describes Apsis as a toolkit for hyperparameter optimization for machine learning algorithms, states its goal is to provide a flexible, simple, and scalable approach, and mentions it is open-sourced under the MIT license. A 'State' section notes that the current version is pre-alpha with limited functionality and documentation.

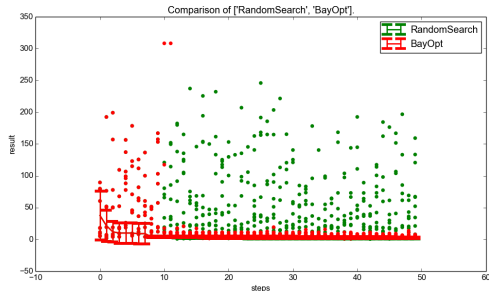
- builds on every commit

Visit **`http://apsis.readthedocs.org`** !

BRANIN HOO OPTIMIZATION



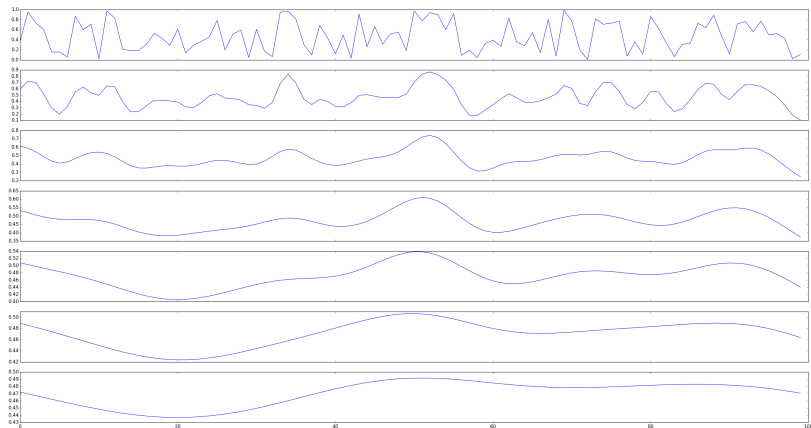
Best Value by Optimizer



Values by Optimizer

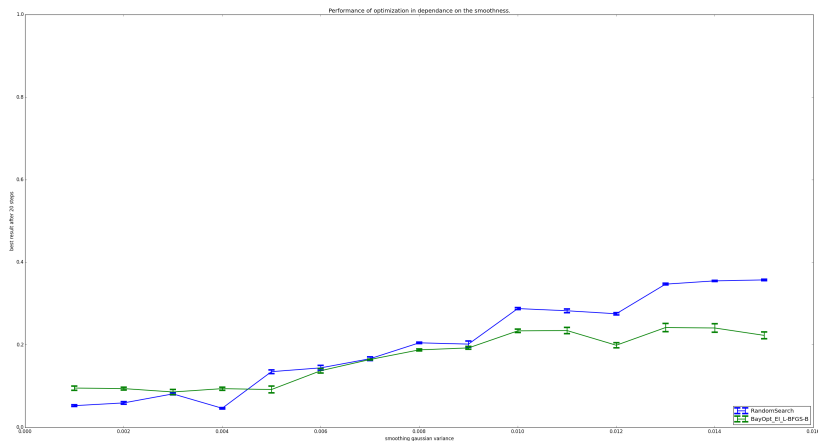
- ▶ random search finds better end result but bay opt is more stable
- ▶ similar performance as in other bay opt literature
- ▶ no other group publishes comparison to random search

OPTIMIZING ARTIFICIAL NOISE FUNCTION



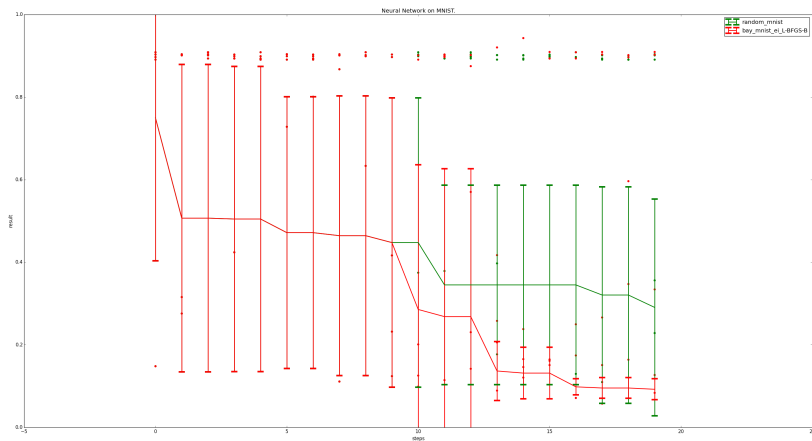
One dimensional noise function with several smoothing variances

OPTIMIZING ARTIFICIAL NOISE FUNCTION



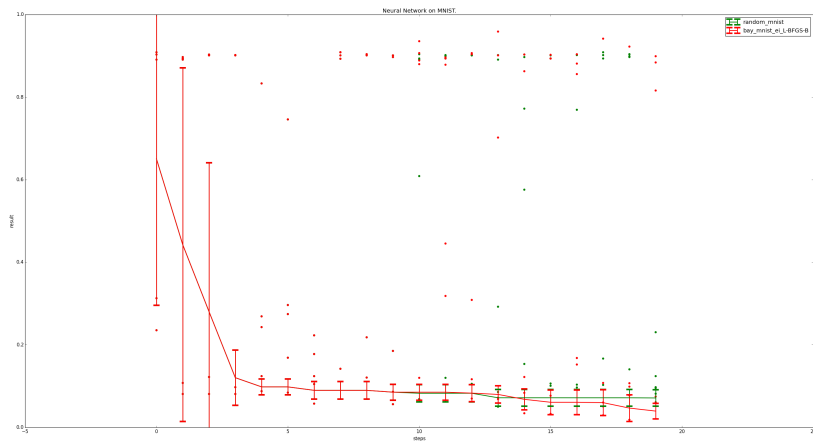
Minimization result on 3d noise by smoothing factor.

BREZE MNIST NEURAL NETWORK



Neural Network on MNIST using uniform parameters

BREZE MNIST NEURAL NETWORK (2)



Neural Network on MNIST using asymptotic parameters for learning rate and learning rate decay

TAKING THE PROJECT TO THE NEXT LEVEL - PROGRAM

- ▶ implement full multicore support
- ▶ implement a REST web-service to offer interoperability with any language
- ▶ improve integration of matplotlib

TAKING THE PROJECT TO THE NEXT LEVEL - BAYESIAN OPTIMIZATION

- ▶ deal with nominal parameters
- ▶ try replacing GPs with Student-t processes
- ▶ try to take tree structured configuration space into account
- ▶ account for evaluation cost depending on hyperparameter setting
- ▶ implement freeze-thaw optimization idea [2]
- ▶ automated learning of input warping [1]

Thank You!

REFERENCES I



Jasper Snoek, Kevin Swersky, Richard S Zemel, and Ryan P Adams.

Input warping for bayesian optimization of non-stationary functions.

arXiv preprint arXiv:1402.0929, 2014.



Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams.
Freeze-thaw bayesian optimization.

arXiv preprint arXiv:1406.3896, 2014.