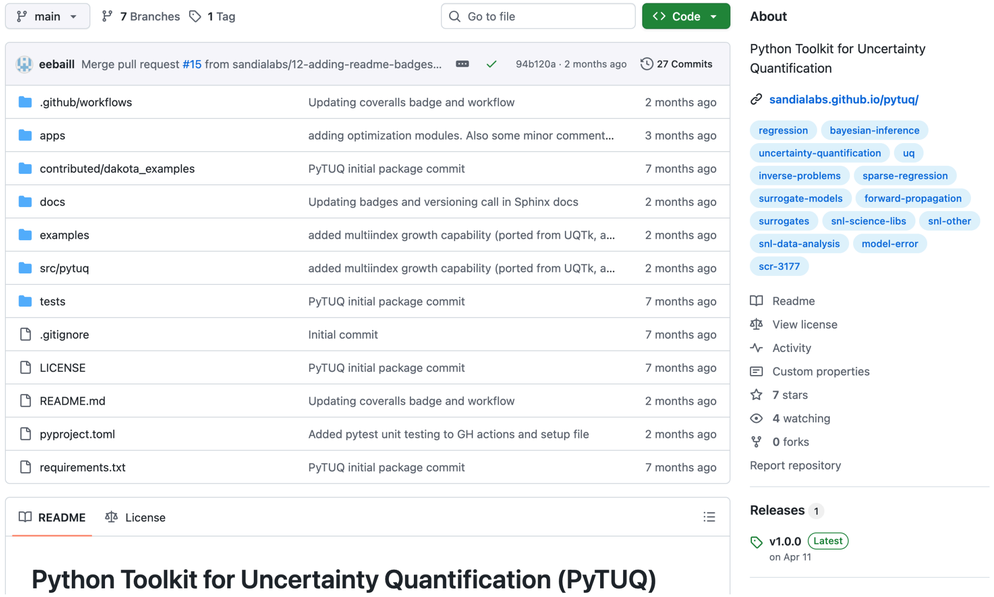
PyTUQ: Python Toolkit for Uncertainty Quantification

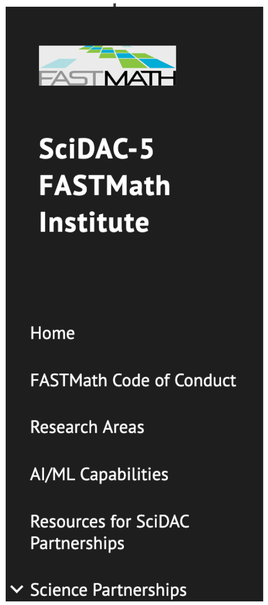
https://github.com/sandialabs/pytuq

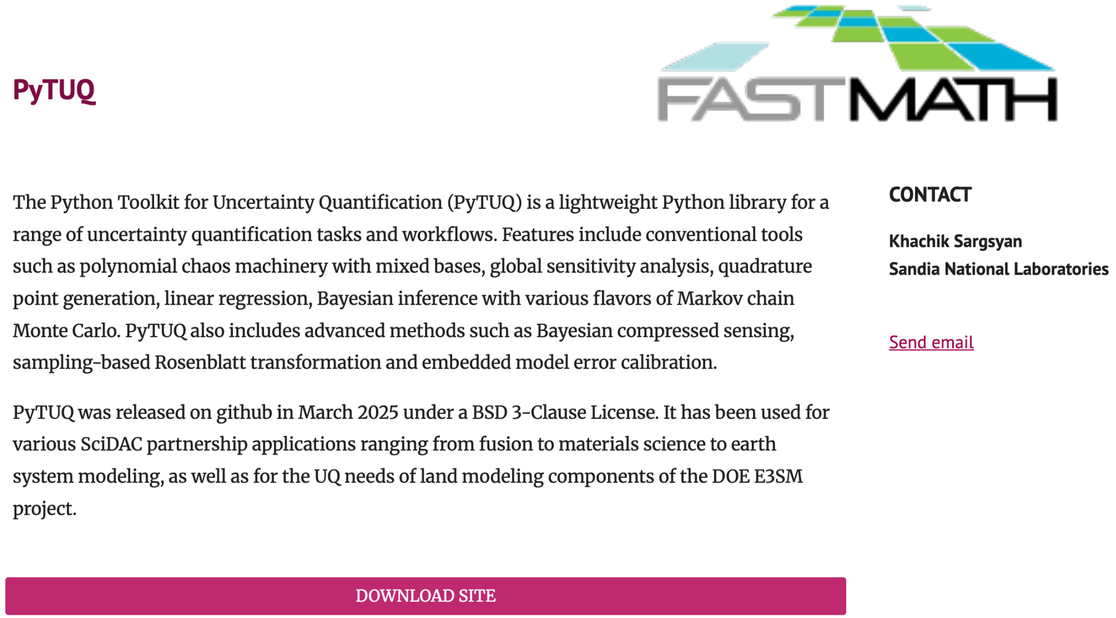
Authors: Khachik Sargsyan, Bert Debusschere, Emilie Grace Baillo

Dependencies: numpy, scipy, matplotlib, (pytorch, dill, quinn)

License: BSD 3-Clause License







How PyTUQ was born

- Had a need and started coding a mixed-bases PC. Later KL kicked in…
  hence the pre-release name KLPC.
- Collaborators requesting little pieces of UQ codes here and there.
- Increasingly more Python usage and less willingness to cmake.
- My own allergy for rewriting the same code twice:
  if I needed to, it was better to write a library.
- Not intended to comprehensively cover all methods:
  add functionality as needed.
- All key UQ functionalities written with OOP principles, with base classes:
  (supposed to be) easily extendable.

- Released March 2025
- Part of SciDAC-6 FASTMath Software catalogue

Current usage

- BER: E3SM Land Model UQ
- BER Partnership: Quasi-biennal Oscillation
- BES: Exascale Catalytic Chemistry
- FES Partnership: ThermChem-FW
- Interlab LDRD: Radiography UQ
- ....

Reduced-dimensional surrogate, calibration, embedded model error, GSA

Stochastic surrogate, GSA, covariance estimation

Linear regression, Uncertainty propgation, GSA

Optimization, Inference/MCMC

Overall Structure

Outer-loop utilities

- Tests
- Examples
- Apps
- Contributed

Core library packages

- func
- rv
- fit
- lreg
- linred
- optim
- minf
- utils
- workflows
- surrogates

Docs

- *https://sandialabs.github.io/pytuq/*
- (Almost) all commented with docstrings
- Mathy details lacking
- Documentation in progress

Overall Structure

Outer-loop utilities

- Tests
- Examples
- Apps
- Contributed

Core library packages

- func
- rv
- fit
- lreg
- linred
- optim
- minf
- utils
- workflows
- surrogates

Base classes
Extendable

Docs

- *https://sandialabs.github.io/pytuq/*
- (Almost) all commented with docstrings
- Mathy details lacking
- Documentation in progress

Tests

Tests with minimal sanity checks

```
test_composefcn.py
test_diam.py
test_dom.py
test_gmm.py
test_hess.py
test_mi.py
```

- Meant to be run out of the box
- *pytest* which automatically runs all *test_*.py* files
- Produces no screen output, only assertion checks
- Supposed to have *test_*.py* for any new capability

  (except I gravitate toward example *ex_*.py* instead of a test)
- Not too many currently

Examples

Examples of varying complexity

ex_bcs.py*	Bayesian Compressive Sensing
ex_gp.py	Gaussian Process regression
ex_gsa.py*	Global Sensitivity Analysis
ex_*ros.py*	(Inverse) Rosenblatt transform
ex_kl.py*	Karhunen-Loève decomposition
⋮	⋮
ex_lreg.py*	Linear regression
ex_mcmc.py*	Markov chain Monte Carlo
ex_minf.py*	Embedded model error inference
ex_pcrv.py	PC multivariate random variable
ex_uprop.py	Uncertainty propagation
⋮	

Examples

Examples of varying complexity

- ex_evidence.py
- ex_func*.py
- ex_integrate.py
- ex_genz1d.py
- ex_mindex*.py
- ⋮
- ex_optim.py
- ex_mixture.py
- ex_quad.py
- ex_sampling.py
- ex_slice.py
- ⋮

- Meant to be run out of the box
- Usually refer to one of these if requests come
- … or write a quick one per request
- Produce screen outputs, and *.png* (and some *.txt*) files
- Script to run all examples together

run_examples.sh

```
#!/bin/bash -e

mkdir -p runex
cd runex

SCRDIR=$(dirname "$0")


echo $SCRDIR
for ex_scr in "$SCRDIR"/../examples/*py; do
    echo "=========================================="
    start=$SECONDS
    ex_scr_name=$(basename "$ex_scr" .py)
    echo "Running  $ex_scr"
    "$ex_scr" > ${ex_scr_name}.log
    duration=$(( SECONDS - start ))
    echo "It took $duration sec."
    #echo "==========================================="
done
```

Apps

Apps and workflows

- *kl_fit.py*, *klsurr_fit.py* — KL fit given data; KL+PC/NN surrogate
- *wf_uqpc.x*, *pc_prep.py*, *pc_sam.py*, *pc_fit.py* — Workflow for PC-based propagation
- *nn_fit.py* — NN surrogate. Quick and dirty. Relies on QUiNN library *https://github.com/sandialabs/quinn*
- *mrun.py* — Multithreading app: Creates folders and runs a list of command lines
- *plot_cov.py*, *plot_pcoord.py*, *plot_pdfs.py*, *plot_xx.py*, *plot_yx.py* — Plotting apps

- Typically command-line arguments and works as-is.
- Similar to command line ‘apps’ in UQTk.
- A few more we currently have outside PyTUQ, waiting to be documented.

Contributed

Contributed packages/modules

- Dakota wrappers
- Room for external contributions

Core library packages

src

- func — Multivariate functions $f : \mathbb{R}^d \to \mathbb{R}^o$
- rv — Multivariate random variables (PCRV, GMM, …)
- lreg — Linear regression (ANL, LSQ, BCS, …)
- fit — Nonlinear fits (GP)
- gsa — Global sensitivity analysis (Sobol, PC, MOAT, …)
- linred — Linear dimensionality reduction (PCA, KLE, …)
- optim — Optimization (GD)
- minf — Model inference (Bayes/MCMC/Embedded Model error)

utils — Plotting, data manipulation, …

workflows — Utilities chaining a few tasks

surrogates — Surrogate wrappers

func

Function package

```
func.py

benchmark.py
genz.py
toy.py
poly.py

chem.py

oper.py
```

- Parent function class
- Multivariate, multioutput functions $f : \mathbb{R}^d \to \mathbb{R}^o$
- Lots of benchmark functions
- Gradients/Hessians
- Operations on functions
- Plotting routines of various slices
- See *ex_func.py* and *ex_funcgrad.py*

```
Fcn1 = benchmark.Ishigami()
Fcn2 = benchmark.Branin()
Fcn = Fcn1 * Fcn2
Fcn.grad(x)
Fcn.plot_2d(d1, d2)
```

```
pcrv = PCRV(nout, ndim, pctype)
[pcfit via e.g. linear regression]
pcrv.setFunction()

newfcn = pcrv.function + benchmark.Ishigami()
```

func

Function package:

side story

- My son Vahan kept bugging me for a home summer code/math project.

func

Function package:

side story

- My son Vahan kept bugging me for a home summer code/math project.
- I asked him to look at the base class and a few examples currently in PyTUQ….
- …. and implement common benchmark functions

func

Function package:

side story

- My son Vahan kept bugging me for a home summer code/math project.
- I asked him to look at the base class and a few examples currently in PyTUQ….
- …. and implement common benchmark functions
- See, e.g. https://infinity77.net/global_optimization/ and https://www.sfu.ca/~ssurjano/
- Also compute by hand and implement analytical gradients (win-win for a father).

func

Function package:

side story

- My son Vahan kept bugging me for a home summer code/math project.
- I asked him to look at the base class and a few examples currently in PyTUQ….
- …. and implement common benchmark functions
- See, e.g. https://infinity77.net/global_optimization/ and https://www.sfu.ca/~ssurjano/
- Also compute by hand and implement analytical gradients (win-win for a father).
- **After a few (4-5)**, he lost motivation and approached me to ask for money.

func

Function package:

side story

- My son Vahan kept bugging me for a home summer code/math project.
- I asked him to look at the base class and a few examples currently in PyTUQ….
- …. and implement common benchmark functions
- See, e.g. https://infinity77.net/global_optimization/ and https://www.sfu.ca/~ssurjano/
- Also compute by hand and implement analytical gradients (win-win for a father).
- **After a few (4-5)**, he lost motivation and approached me to ask for money.
- We converged at $10/function and the money goes to his next guitar purchase…

  (again, a win-win for a father, and money laundering of sorts).
- … including gradients and documentation python/sphinx/latex style, and of course, tests

func

Function package:

side story

- My son Vahan kept bugging me for a home summer code/math project.
- I asked him to look at the base class and a few examples currently in PyTUQ….
- …. and implement common benchmark functions
- See, e.g. https://infinity77.net/global_optimization/ and https://www.sfu.ca/~ssurjano/
- Also compute by hand and implement analytical gradients (win-win for a father).
- **After a few (4-5)**, he lost motivation and approached me to ask for money.
- We converged at $10/function and the money goes to his next guitar purchase…

  (again, a win-win for a father, and money laundering of sorts).
- … including gradients and documentation python/sphinx/latex style, and of course, tests
- **After a few more (8-10)**, he got bored again

  (he probably thought he should have asked for more money).

func

Function package:

side story

- My son Vahan kept bugging me for a home summer code/math project.
- I asked him to look at the base class and a few examples currently in PyTUQ….
- …. and implement common benchmark functions
- See, e.g. https://infinity77.net/global_optimization/ and https://www.sfu.ca/~ssurjano/
- Also compute by hand and implement analytical gradients (win-win for a father).
- **After a few (4-5)**, he lost motivation and approached me to ask for money.
- We converged at $10/function and the money goes to his next guitar purchase…

  (again, a win-win for a father, and money laundering of sorts).
- … including gradients and documentation python/sphinx/latex style, and of course, tests
- **After a few more (8-10)**, he got bored again

  (he probably thought he should have asked for more money).
- So he wrote a Latex-to-Python interpreter/converter (took a couple of weeks)

that auto-writes the PyTUQ code given a latex string of the formula.

func

Function package: side story

https://github.com/Vahanosi4ek

github.com/Vahanosi4ek/pytuq_funcs/blob/main/benchmark/benchmark.py

Files

main

Go to file

- __pycache__
- benchmark
  - __pycache__
  - benchmark.py
  - benchmark_nn.py
  - benchmark_nn_test.py
  - benchmark_test.py
- latex_compiler
  - __pycache__
  - autograd.py
  - codegen.py
  - function_creator.py
  - grammar.txt
  - lexer.py
  - main.py
  - nodes.py
  - optimizer.py
  - parser.py
  - tree_manip.py
- .gitignore
- LICENSE
- README.md

pytuq_funcs / benchmark / benchmark.py

Vahanosi4ek more funcs

Code Blame 3440 lines (2535 loc) · 127 KB

```python
#!/usr/bin/env python3

import numpy as np
from pytuq.func.func import Function

# TODO: add complete support for discontinuous and nondifferentiable functions

# https://infinity77.net/global_optimization/test_functions.html#test-function
# 1-d functions

class SineSum(Function):
    """
    Problem 02 [https://infinity77.net/global_optimization/test_functions_1d.h
    """
    def __init__(self, c1=1., c2=10/3, name="SineSum"):
        super().__init__()
        self.name = name
        self.c1, self.c2 = c1, c2
        self.dim = 1
        self.outdim = 1

        self.setDimDom(domain=np.ones((self.dim, 1)) * np.array([2.7, 7.5]))

    def __call__(self, x):
        r"""Simple sum of sines

        .. math::
            f(x)=\sin(c_1x)+\sin(c_2x)


        Default constant values are :math:`c = (1., 10/3)`.

        Args:
            x (np.ndarray): Input array :math:`x` of size `(N,1)`.

        Returns:
            np.ndarray: Output array of size `(N,1)`.
        """
        return np.sin(self.c1 * x) + np.sin(self.c2 * x)

    def grad(self, x):
```

func

Function package: side story

https://github.com/Vahanosi4ek

github.com/Vahanosi4ek/pytuq_funcs/blob/main/benchmark/benchmark.py

Files

main

Go to file

- __pycache__
- benchmark
  - __pycache__
  - benchmark.py
  - benchmark_nn.py
  - benchmark_nn_test.py
  - benchmark_test.py
- latex_compiler
  - __pycache__
  - autograd.py
  - codegen.py
  - function_creator.py
  - grammar.txt
  - lexer.py
  - main.py
  - nodes.py
  - optimizer.py
  - parser.py
  - tree_manip.py
- .gitignore
- LICENSE
- README.md

pytuq_funcs / benchmark / benchmark.py

Vahanosi4ek more funcs

Code Blame 3440 lines (2535 loc) · 127 KB

```python
#!/usr/bin/env python3

import numpy as np
from pytuq.func.func import Function

# TODO: add complete support for discontinuous and nondifferentiable functions

# https://infinity77.net/global_optimization/test_functions.html#test-function
# 1-d functions

class SineSum(Function):
    """
    Problem 02 [https://infinity77.net/global_optimization/test_functions_1d.h
    """
    def __init__(self, c1=1., c2=10/3, name="SineSum"):
        super().__init__()
        self.name = name
        self.c1, self.c2 = c1, c2
        self.dim = 1
        self.outdim = 1

        self.setDimDom(domain=np.ones((self.dim, 1)) * np.array([2.7, 7.5]))

    def __call__(self, x):
        r"""Simple sum of sines

        .. math::
            f(x)=\sin(c_1x)+\sin(c_2x)


        Default constant values are :math:`c = (1., 10/3)`.

        Args:
            x (np.ndarray): Input array :math:`x` of size `(N,1)`.

        Returns:
            np.ndarray: Output array of size `(N,1)`.
        """
        return np.sin(self.c1 * x) + np.sin(self.c2 * x)

    def grad(self, x):
```

```python
# N-D test functions, alphabe
    Ackley(),
    Adjiman(),
    Alpine01(),
    Alpine02(),
    AMGM(),
    BartelsConn(),
    Bird(),
    Bohachevsky(),
    Branin01(),
    Branin02(),
    Brent(),
    Bukin02(),
    Bukin04(),
    Bukin6(),
    CarromTable(),
    Chichinadze(),
    Cigar(),
    Colville(),
    CosineMixture(),
    Damavandi(),
    DeckkersAarts(),
    Dolan(),
    EggCrate(),
    ElAttarVidyasagarDutta(),
    FreudensteinRoth(),
    GoldsteinPrice(),
    HimmelBlau(),
    Hosaki(),
    # Keane(), X
    Leon(),
    Levy13(),
    Matyas(),
    McCormick(),
    # MieleCantrell(), X
    # Mishra03(), X
    # Mishra04(), X
    Mishra05(),
    Mishra06(),
    # Mishra08(), X
    NewFunction03(),
```

```python
    Parsopoulos(),
    Powell(),
    Price01(),
    Price02(),
    Price03(),
    Price04(),
    Quadratic(),
    RosenbrockModified(),
    RotatedEllipse01(),
    RotatedEllipse02(),
    Schaffer01(),
    Schaffer02(),
    Schaffer04(),
    # SchmidtVetters(), X
    Schwefel36(),
    SixHumpCamel(),
    ThreeHumpCamel(),
    Treccani(),
    Trefethen(),
    Ursem01(),
    Ursem03(),
    # Ursem04(), X
    UrsemWaves(),
    VenterSobiezcczanskiSobie
    WayburnSeader01(),
    WayburnSeader02(),
    Wolfe(),
    Zettl(),
    Zirilli(),

# Many local minima (it's wor
    CrossInTray(),
    DropWave(),
    EggHolder(),
    Griewank(),

# Trig
    ChengSandu(),
    Sine1d(),
    Forrester(),
    Friedman(),
    GramacyLee(),
    GramacyLee2(),
```

func

Function package: side story

https://github.com/Vahanosi4ek

github.com/Vahanosi4ek/pytuq_funcs/blob/main/benchmark/benchmark.py

Files

main

Go to file

- __pycache__
- benchmark
  - __pycache__
  - benchmark.py
  - benchmark_nn.py
  - benchmark_nn_test.py
  - benchmark_test.py
- latex_compiler
  - __pycache__
  - autograd.py
  - codegen.py
  - function_creator.py
  - grammar.txt
  - lexer.py
  - main.py
  - nodes.py
  - optimizer.py
  - parser.py
  - tree_manip.py
- .gitignore
- LICENSE
- README.md

pytuq_funcs / benchmark / benchmark.py

Vahanosi4ek more funcs

Code Blame 3440 lines (2535 loc) · 127 KB

```python
#!/usr/bin/env python3

import numpy as np
from pytuq.func.func import Function

# TODO: add complete support for discontinuous and nondifferentiable functions

# https://infinity77.net/global_optimization/test_functions.html#test-function
# 1-d functions

class SineSum(Function):
    """
    Problem 02 [https://infinity77.net/global_optimization/test_functions_1d.h
    """
    def __init__(self, c1=1., c2=10/3, name="SineSum"):
        super().__init__()
        self.name = name
        self.c1, self.c2 = c1, c2
        self.dim = 1
        self.outdim = 1

        self.setDimDom(domain=np.ones((self.dim, 1)) * np.array([2.7, 7.5]))

    def __call__(self, x):
        r"""Simple sum of sines

        .. math::
            f(x)=\sin(c_1x)+\sin(c_2x)


        Default constant values are :math:`c = (1., 10/3)`.

        Args:
            x (np.ndarray): Input array :math:`x` of size `(N,1)`.

        Returns:
            np.ndarray: Output array of size `(N,1)`.
        """
        return np.sin(self.c1 * x) + np.sin(self.c2 * x)

    def grad(self, x):
```

He now has about 100 functions, ready to be forked and merged to PyTUQ. Will do soon. School just started.

```python
# N-D test functions, alphabe
    Ackley(),
    Adjiman(),
    Alpine01(),
    Alpine02(),
    AMGM(),
    BartelsConn(),
    Bird(),
    Bohachevsky(),
    Branin01(),
    Branin02(),
    Brent(),
    Bukin02(),
    Bukin04(),
    Bukin6(),
    CarromTable(),
    Chichinadze(),
    Cigar(),
    Colville(),
    CosineMixture(),
    Damavandi(),
    DeckkersAarts(),
    Dolan(),
    EggCrate(),
    ElAttarVidyasagarDutta(),
    FreudensteinRoth(),
    GoldsteinPrice(),
    HimmelBlau(),
    Hosaki(),
    # Keane(), X
    Leon(),
    Levy13(),
    Matyas(),
    McCormick(),
    # MieleCantrell(), X
    # Mishra03(), X
    # Mishra04(), X
    Mishra05(),
    Mishra06(),
    # Mishra08(), X
    NewFunction03(),
    Parsopoulos(),
    Powell(),
    Price01(),
    Price02(),
    Price03(),
    Price04(),
    Quadratic(),
    RosenbrockModified(),
    RotatedEllipse01(),
    RotatedEllipse02(),
    Schaffer01(),
    Schaffer02(),
    Schaffer04(),
    # SchmidtVetters(), X
    Schwefel36(),
    SixHumpCamel(),
    ThreeHumpCamel(),
    Treccani(),
    Trefethen(),
    Ursem01(),
    Ursem03(),
    # Ursem04(), X
    UrsemWaves(),
    VenterSobiezcczanskiSobie
    WayburnSeader01(),
    WayburnSeader02(),
    Wolfe(),
    Zettl(),
    Zirilli(),

# Many local minima (it's wor
    CrossInTray(),
    DropWave(),
    EggHolder(),
    Griewank(),

# Trig
    ChengSandu(),
    Sine1d(),
    Forrester(),
    Friedman(),
    GramacyLee(),
    GramacyLee2(),
```

func

Function package: side story

https://github.com/Vahanosi4ek

github.com/Vahanosi4ek/pytuq_funcs/blob/main/benchmark/benchmark.py

Files

main

Go to file

- __pycache__
- benchmark
  - __pycache__
  - benchmark.py
  - benchmark_nn.py
  - benchmark_nn_test.py
  - benchmark_test.py
- latex_compiler
  - __pycache__
  - autograd.py
  - codegen.py
  - function_creator.py
  - grammar.txt
  - lexer.py
  - main.py
  - nodes.py
  - optimizer.py
  - parser.py
  - tree_manip.py
- .gitignore
- LICENSE
- README.md

pytuq_funcs / benchmark / benchmark.py

Vahanosi4ek more funcs

Code Blame 3440 lines (2535 loc) · 127 KB

```python
#!/usr/bin/env python3

import numpy as np
from pytuq.func.func import Function

# TODO: add complete support for discontinuous and nondifferentiable functions

# https://infinity77.net/global_optimization/test_functions.html#test-function
# 1-d functions

class SineSum(Function):
    """
    Problem 02 [https://infinity77.net/global_optimization/test_functions_1d.h
    """
    def __init__(self, c1=1., c2=10/3, name="SineSum"):
        super().__init__()
        self.name = name
        self.c1, self.c2 = c1, c2
        self.dim = 1
        self.outdim = 1

        self.setDimDom(domain=np.ones((self.dim, 1)) * np.array([2.7, 7.5]))

    def __call__(self, x):
        r"""Simple sum of sines

        .. math::
            f(x)=\sin(c_1x)+\sin(c_2x)


        Default constant values are :math:`c = (1., 10/3)`.

        Args:
            x (np.ndarray): Input array :math:`x` of size `(N,1)`.

        Returns:
            np.ndarray: Output array of size `(N,1)`.
        """
        return np.sin(self.c1 * x) + np.sin(self.c2 * x)

    def grad(self, x):
```

He now has about 100 functions, ready to be forked and merged to PyTUQ. Will do soon. School just started.

I paid with bonus, we brought back a $3k guitar from Spain:)

```python
# N-D test functions, alphabe
    Ackley(),
    Adjiman(),
    Alpine01(),
    Alpine02(),
    AMGM(),
    BartelsConn(),
    Bird(),
    Bohachevsky(),
    Branin01(),
    Branin02(),
    Brent(),
    Bukin02(),
    Bukin04(),
    Bukin6(),
    CarromTable(),
    Chichinadze(),
    Cigar(),
    Colville(),
    CosineMixture(),
    Damavandi(),
    DeckkersAarts(),
    Dolan(),
    EggCrate(),
    ElAttarVidyasagarDutta(),
    FreudensteinRoth(),
    GoldsteinPrice(),
    HimmelBlau(),
    Hosaki(),
    # Keane(), X
    Leon(),
    Levy13(),
    Matyas(),
    McCormick(),
    # MieleCantrell(), X
    # Mishra03(), X
    # Mishra04(), X
    Mishra05(),
    Mishra06(),
    # Mishra08(), X
    NewFunction03(),
    Parsopoulos(),
    Powell(),
    Price01(),
    Price02(),
    Price03(),
    Price04(),
    Quadratic(),
    RosenbrockModified(),
    RotatedEllipse01(),
    RotatedEllipse02(),
    Schaffer01(),
    Schaffer02(),
    Schaffer04(),
    # SchmidtVetters(), X
    Schwefel36(),
    SixHumpCamel(),
    ThreeHumpCamel(),
    Treccani(),
    Trefethen(),
    Ursem01(),
    Ursem03(),
    # Ursem04(), X
    UrsemWaves(),
    VenterSobiezcczanskiSobie
    WayburnSeader01(),
    WayburnSeader02(),
    Wolfe(),
    Zettl(),
    Zirilli(),

# Many local minima (it's wor
    CrossInTray(),
    DropWave(),
    EggHolder(),
    Griewank(),

# Trig
    ChengSandu(),
    Sine1d(),
    Forrester(),
    Friedman(),
    GramacyLee(),
    GramacyLee2(),
```

func

Function package: side story

https://github.com/V

github.com/Vahanosi4ek/pytuq_funcs/blob/main/benchmark/ben

Files

main

Go to file

- __pycache__
- benchmark
  - __pycache__
  - benchmark.py
  - benchmark_nn.py
  - benchmark_nn_test.py
  - benchmark_test.py
- latex_compiler
  - __pycache__
  - autograd.py
  - codegen.py
  - function_creator.py
  - grammar.txt
  - lexer.py
  - main.py
  - nodes.py
  - optimizer.py
  - parser.py
  - tree_manip.py
- .gitignore
- LICENSE
- README.md

pytuq_funcs / benchmark / benchmark.py

Vahanosi4ek more funcs

Code Blame 3440 lines (2535 loc)

```python
#!/usr/bin/env python3

import numpy as np
from pytuq.func.func import Fu

# TODO: add complete support f

# https://infinity77.net/globa
# 1-d functions

class SineSum(Function):
    """
    Problem 02 [https://infin
    """
    def __init__(self, c1=1.,
        super().__init__()
        self.name = name
        self.c1, self.c2 = c1,
        self.dim = 1
        self.outdim = 1

        self.setDimDom(domain=

    def __call__(self, x):
        r"""Simple sum of sine

        .. math::
            f(x)=\sin(c_1x)+\s


        Default constant value

        Args:
            x (np.ndarray): In

        Returns:
            np.ndarray: Output array of size `(N,1)`.
        """
        return np.sin(self.c1 * x) + np.sin(self.c2 * x)

    def grad(self, x):
```

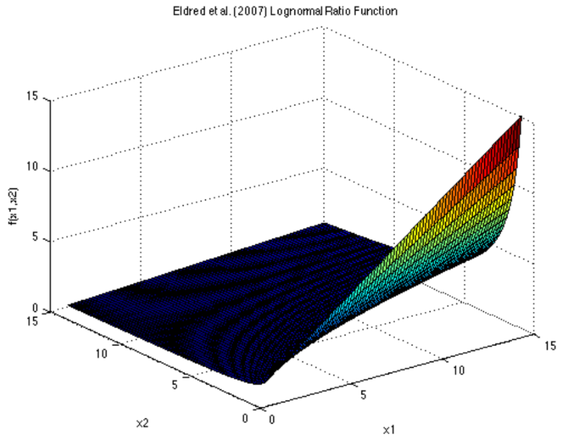
sfu.ca/~ssurjano/eldetal07ratio.html

- HOME
- OPTIMIZATION
- EMULATION/ PREDICTION
- UNCERTAINTY QUANTIFICATION
- MULTI FIDELITY SIMULATION
- CALIBRATION/ TUNING
- SCREENING
- INTEGRATION
- FUNCTIONAL DATA
- ABOUT
- OTHER TEST FUNCTIONS AND CODE

Test Functions and Datasets

Uncertainty Quantification Test Problems

ELDRED ET AL. (2007) LOGNORMAL RATIO FUNCTION



$$f(\mathbf{x}) = \frac{x_1}{x_2}$$

Description:

Dimensions: 2

This test example is a limit state function, defining the boundary between safe and failed region domain (Eldred et al., 2070).

Input Distributions:

The distributions of the input random variables are: $x_i \sim$ Lognormal(1, 0.5), for all i = 1, 2. There

```python
# N-D test functions, alphabe
    Ackley(),
    Adjiman(),
    Alpine01(),
    Alpine02(),
    AMGM(),
    BartelsConn(),
    Bird(),
    Bohachevsky(),
    Branin01(),
    Branin02(),
    Brent(),
    Bukin02(),
    Bukin04(),
    Bukin6(),
    CarromTable(),
    Chichinadze(),
    Cigar(),
    Colville(),
    CosineMixture(),
    Damavandi(),
    DeckkersAarts(),
    Dolan(),
    EggCrate(),
    ElAttarVidyasagarDutta(),
    FreudensteinRoth(),
    GoldsteinPrice(),
    HimmelBlau(),
    Hosaki(),
    # Keane(), X
    Leon(),
    Levy13(),
    Matyas(),
    McCormick(),
    # MieleCantrell(), X
    # Mishra03(), X
    # Mishra04(), X
    Mishra05(),
    Mishra06(),
    # Mishra08(), X
    NewFunction03(),
    Parsopoulos(),
    Powell(),
    Price01(),
    Price02(),
    Price03(),
    Price04(),
    Quadratic(),
    RosenbrockModified(),
    RotatedEllipse01(),
    RotatedEllipse02(),
    Schaffer01(),
    Schaffer02(),
    Schaffer04(),
    # SchmidtVetters(), X
    Schwefel36(),
    SixHumpCamel(),
    ThreeHumpCamel(),
    Treccani(),
    Trefethen(),
    Ursem01(),
    Ursem03(),
    # Ursem04(), X
    UrsemWaves(),
    VenterSobiezcczanskiSobie
    WayburnSeader01(),
    WayburnSeader02(),
    Wolfe(),
    Zettl(),
    Zirilli(),

# Many local minima (it's wor
    CrossInTray(),
    DropWave(),
    EggHolder(),
    Griewank(),

# Trig
    ChengSandu(),
    Sine1d(),
    Forrester(),
    Friedman(),
    GramacyLee(),
    GramacyLee2(),
```

rv

Random variable package

- *mrv.py*
- *pcrv.py*
- *rosen.py*

- Parent random variable class
- Multivariate random variables
- **PCRV**, GMM, MVN, MCMCRV
- Sample, evaluate pdf, cdf
- Operations on random variables: mixture, inverse
-
- Rosenblatt transformation

fit

General fitting package (supervised ML)

fit.py

gp.py

- Parent fit class
- Gaussian process construction
- See *ex_gp.py*

gsa

Global sensitivity analysis package

gsa.py

- Parent gsa class
- Regression-based Sobol
- PC Sobol
- MOAT
- See *ex_gsa*.py*

lreg

Linear regression package

- *lreg.py*
- *anl.py*
- *opt.py*
- *bcs.py*
- *merr.py*

- Parent linear regression class

  (derives from *fit* class)
- Given basis evaluator, or an A-matrix
- *anl* includes variational inference
- *fit(), predict()* similar to Scikit-learn
- *predict()* produces prediction (co)variance, too
- See *ex_lreg*.py*

linred

Linear dimensionalty reduction package

- linred.py
- svd.py
- kle.py
- klsurr.py
- klnn.py

- Parent linear dim. red. class
- Derived SVD and KLE classes
- KL+Surrogate capabilities
- Extensible to nonlinear?

Manifold? Autoencoders?

optim

Optimization package

- *optim.py*
- *gd.py*

- Parent class to allow derived optimization methods.
- Bare-bones gradient descent.
- See *ex_optim.py*
- Need to add a lot more.

minf

Model inference package

- *calib.py*
- *infer.py*
- *minf.py*
- *mcmc.py*
- *likelihoods.py*
- *priors.py*

- Parent calibration class, with a similar signature to optimization class
- Born out of embedded model error work
- Allows flexible PC embedding
- MCMC flavors: AMCMC, HMC, MALA
- See *ex_minf*.py* and *ex_mcmc*.py*

utils

Utilities of all sort (no classes)

- decors.py
- funcbank.py
- maps.py
- metrics.py
- stats.py
- mindex.py
- integr.py
- plotting.py
- uqtk_utils.py
- xutils.py

- These are common utilities used across the library
- Some shared between PyTUQ and QUiNN
- Some re-org needed

workflows

Workflows package (more to come)

fits.py	Module for multi-output fit tasks
uprop.py	Module for uncertainty propagation tasks

surrogates

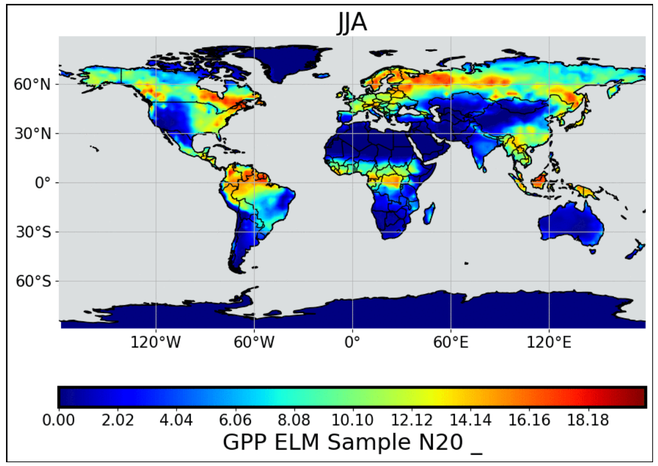
Surrogate wrappers

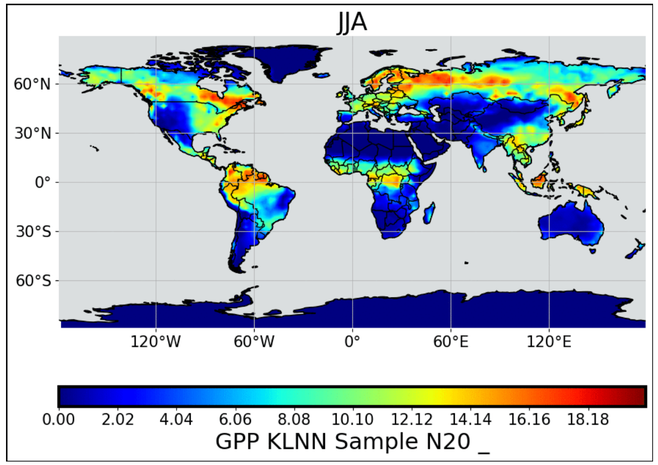
nn.py	Wrapper class to QUiNN's NN surrogate
pce.py	Wrapper class to PyTUQ's PCE surrogate

Application Examples

Reduced dim. surrogate-enabled calibration

BER: E3SM Land Model UQ



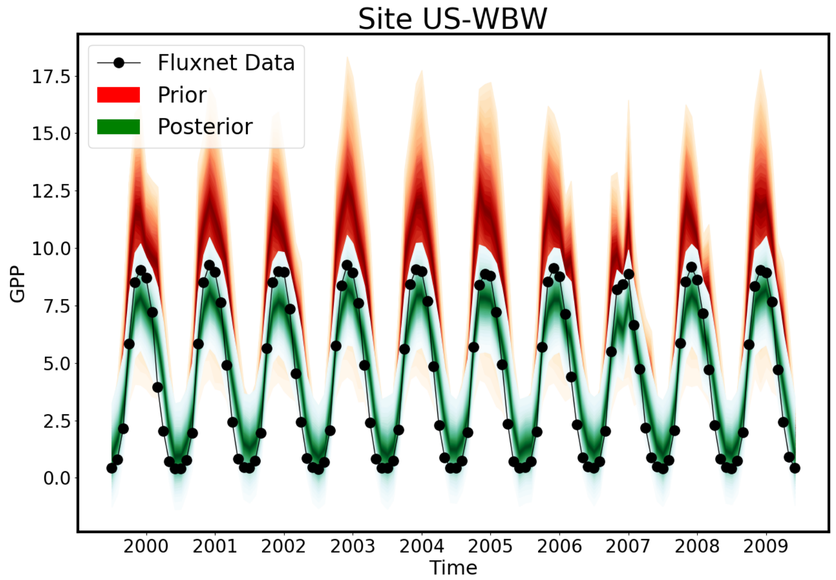


KL+NN spatiotemporal surrogate, 10 params

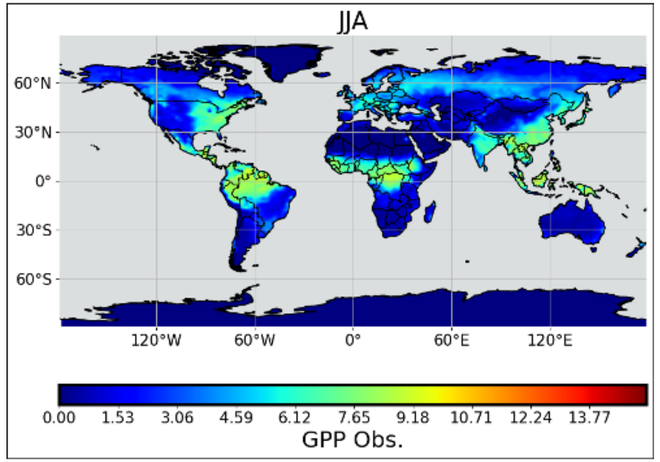
E3SM Energy Exascale Earth System Model

Spatial slice of calibrated model

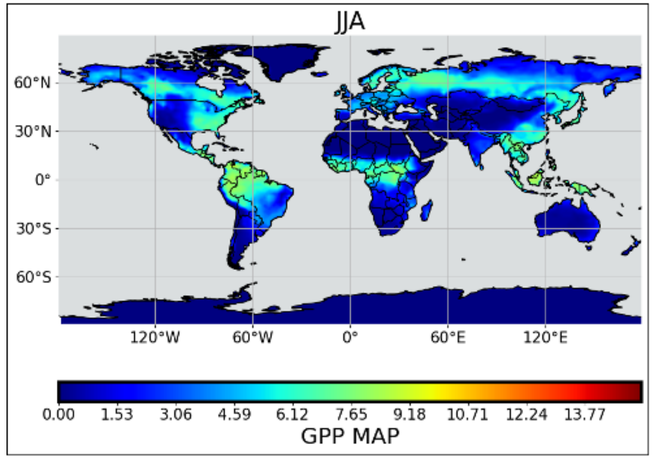
Temporal slice of calibrated model



FLUXCOM Data

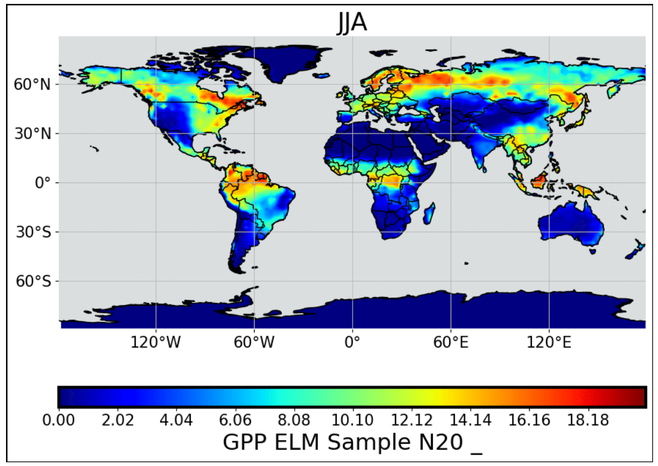


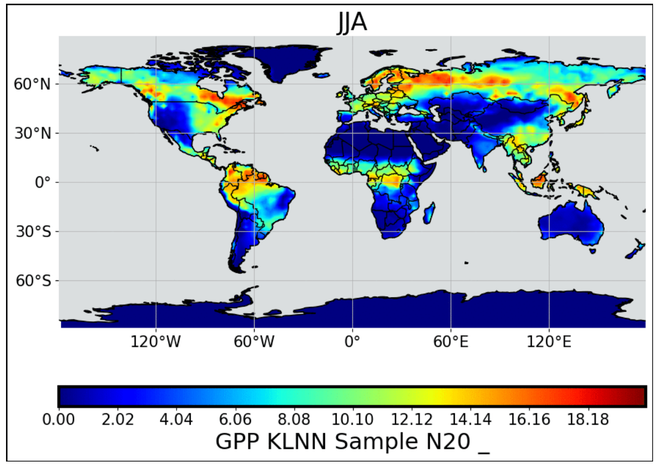
Calibrated Model



Reduced dim. surrogate-enabled calibration

BER: E3SM Land Model UQ



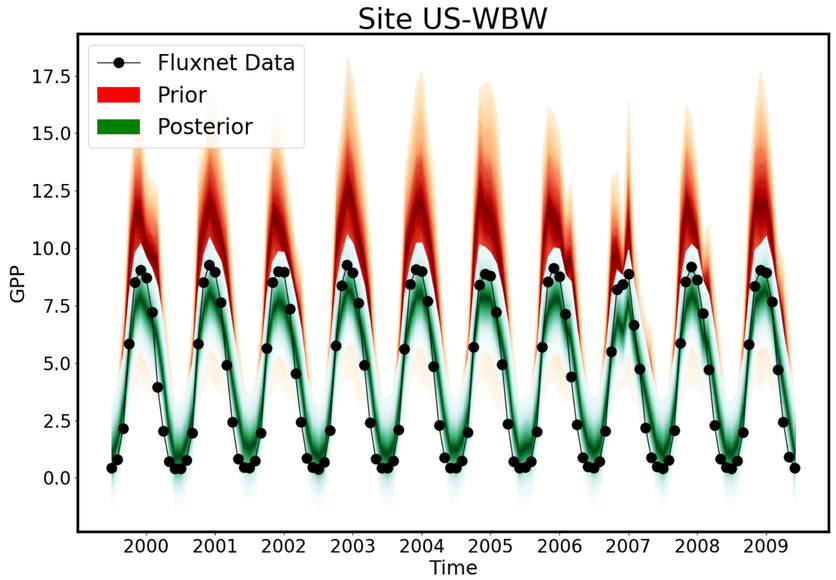


KL+NN spatiotemporal surrogate, 10 params

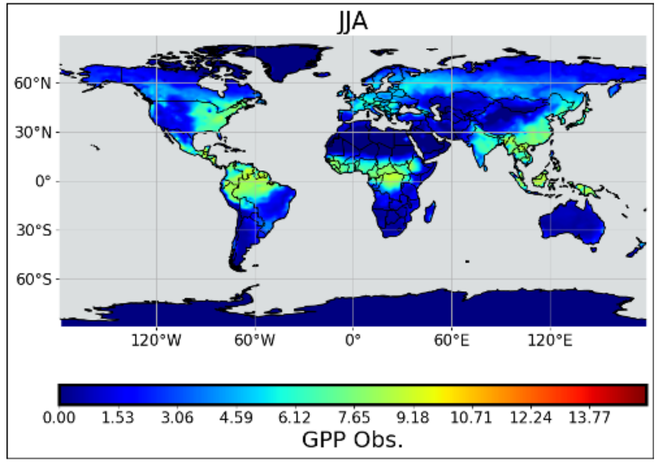
E3SM Energy Exascale Earth System Model

Spatial slice of calibrated model

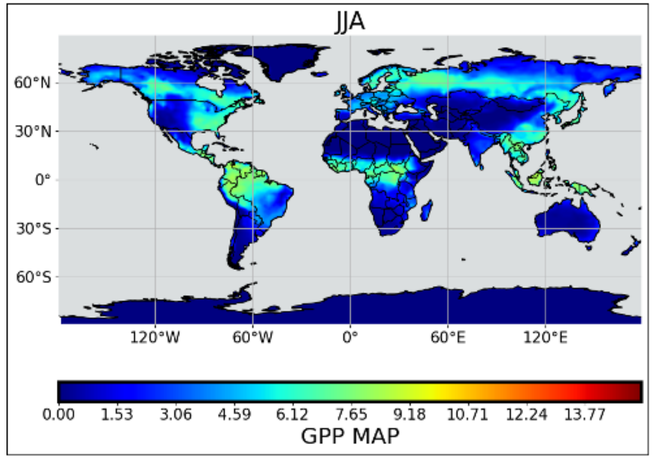
Temporal slice of calibrated model



FLUXCOM Data

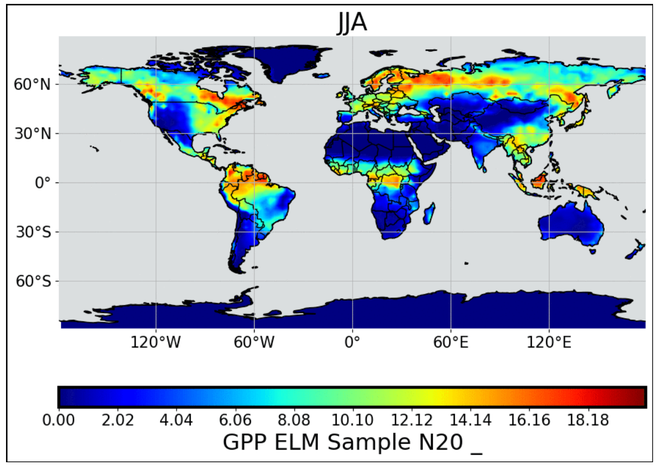


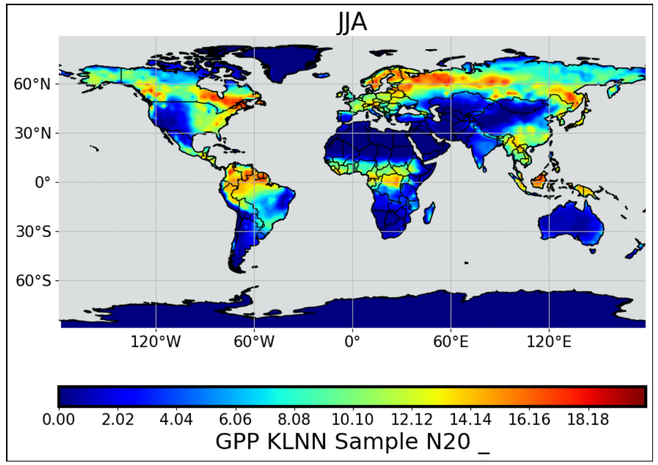
Calibrated Model



Reduced dim. surrogate-enabled calibration

BER: E3SM Land Model UQ



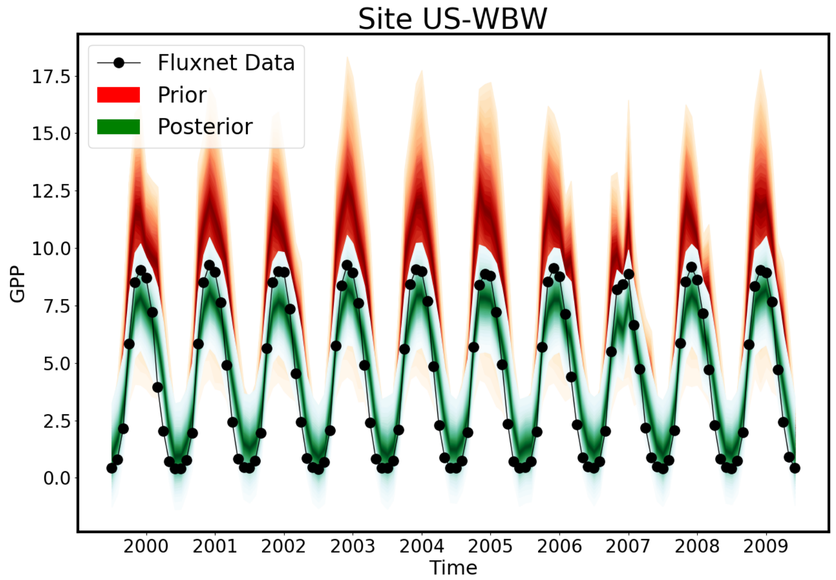


KL+NN spatiotemporal surrogate, 10 params

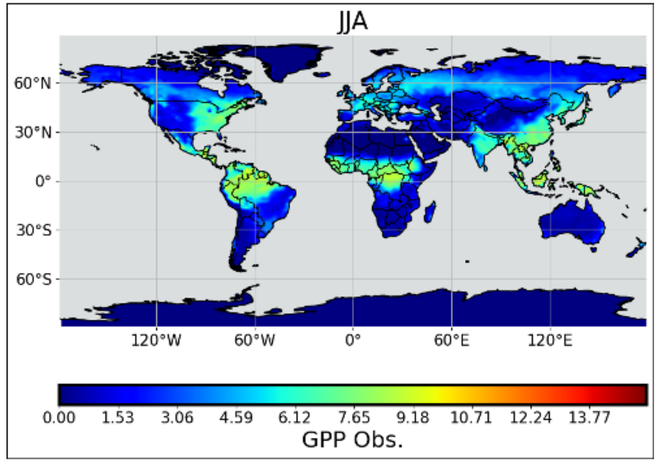
E3SM Energy Exascale Earth System Model

Spatial slice of calibrated model

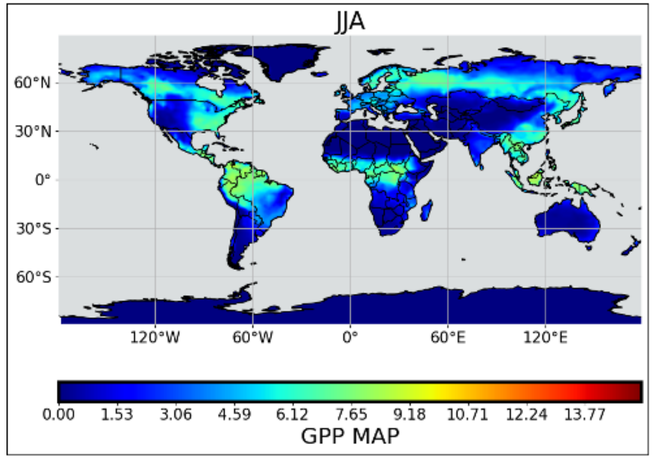
Temporal slice of calibrated model



FLUXCOM Data



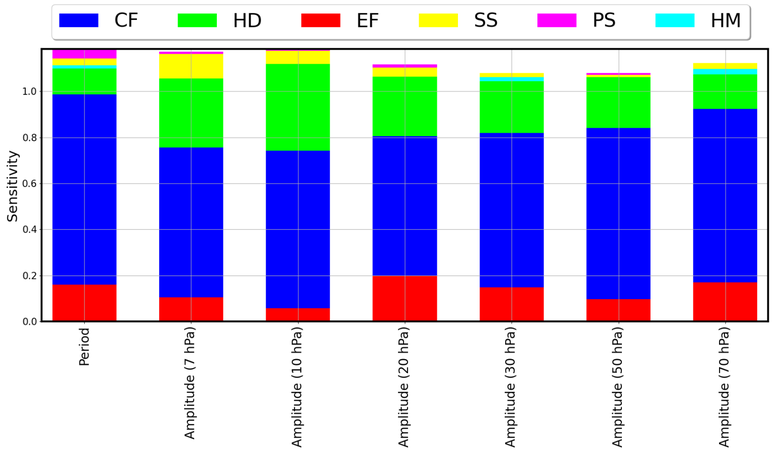
Calibrated Model



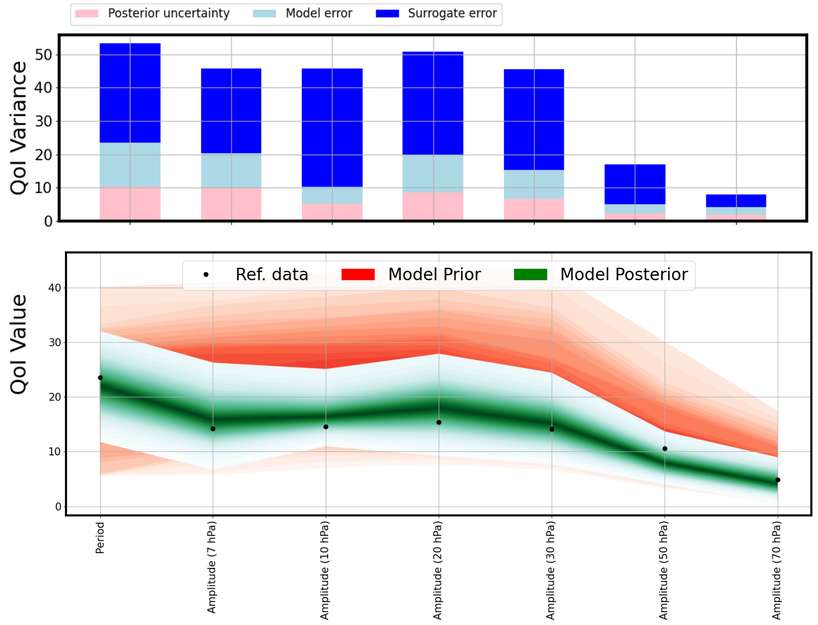
Model error estimation

BER Partnership on QBO

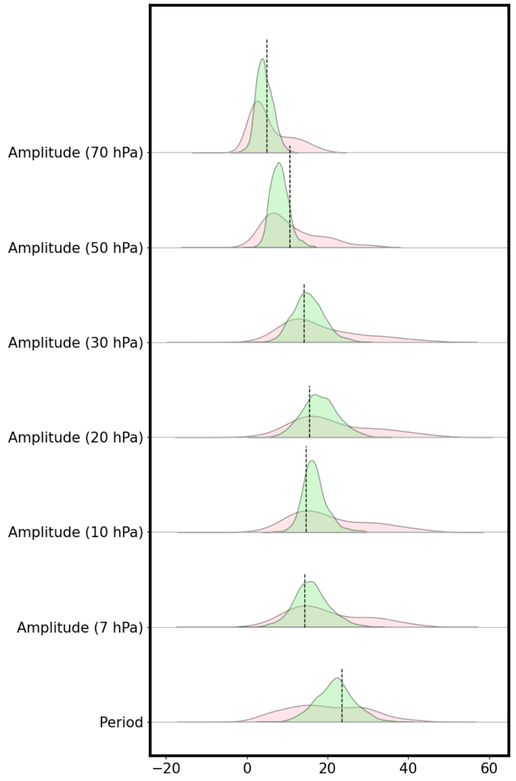
6-param perturbed ensemble, PC surrogate + GSA

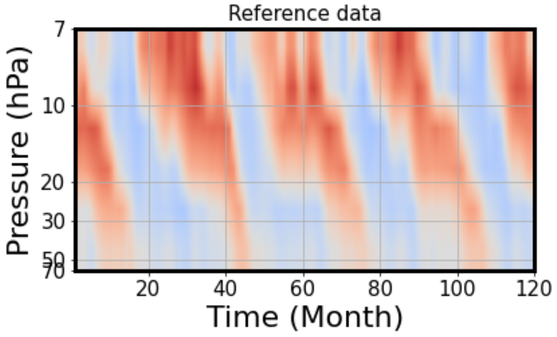


Calibration with model error and uncertainty attribution



Marginal posterior predictives





KL dimensionality reduction of wind fields, KL+PC surrogate to enable calibration

MLIAP Uncertainty estimation

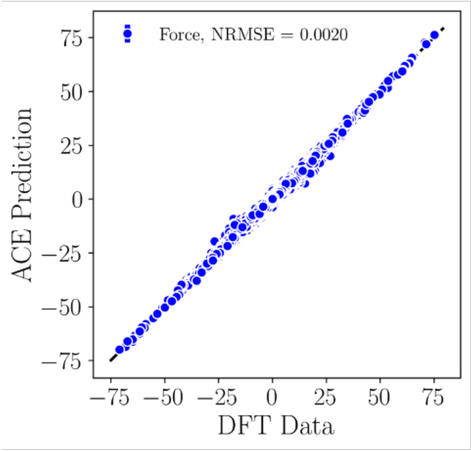
FES Partnership ThermChem-FW

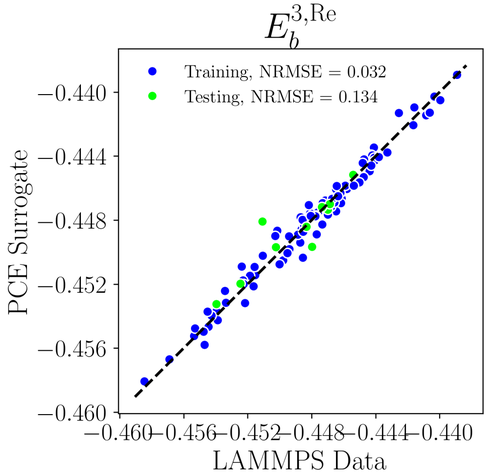
Bayesian linear regression (BLR) to fit, with UQ, interatomic potential to DFT data

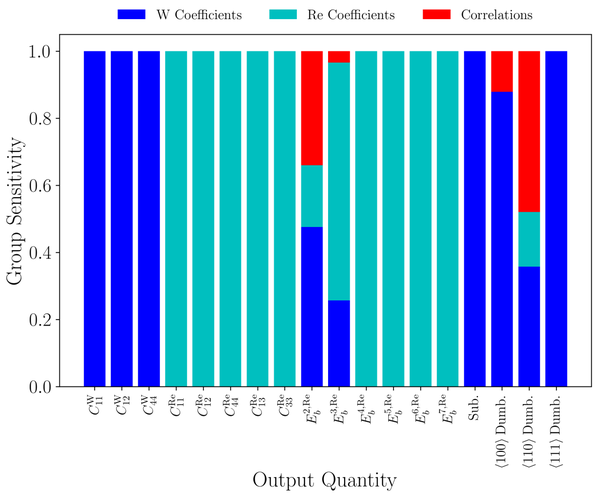
PC-based propagation of uncertainty through MD QoIs

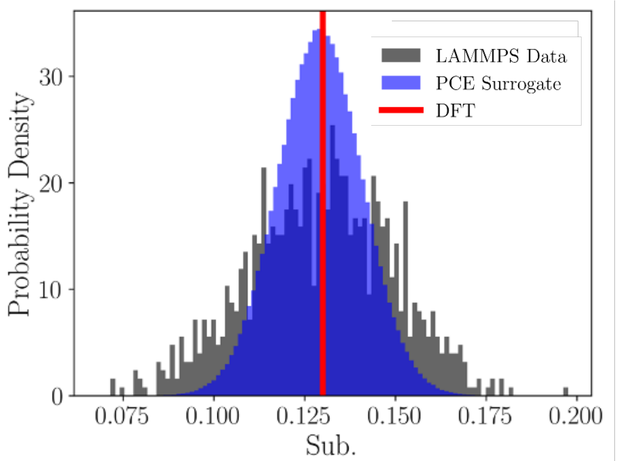
GSA and uncertainty attribution

Validation with DFT data





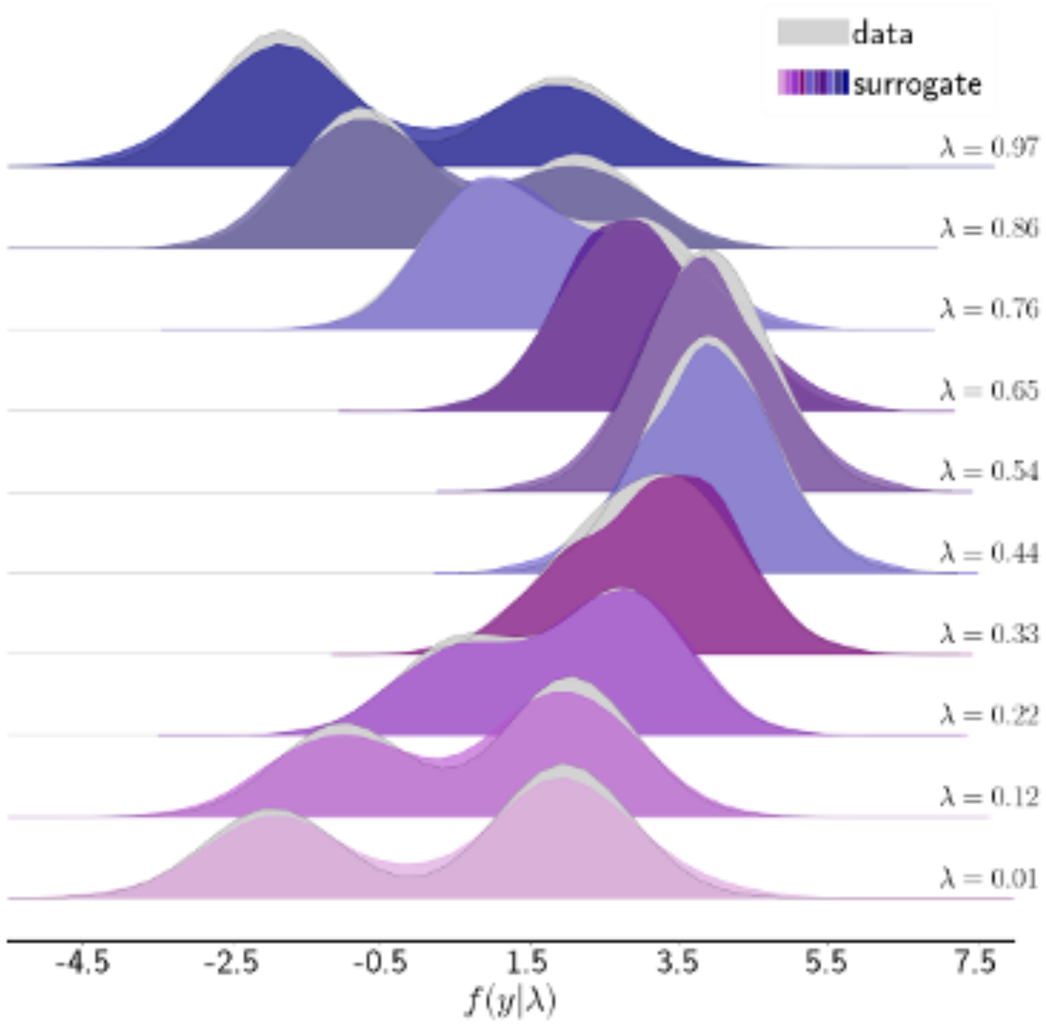




Figs courtesy of Cristian Lacey

Parametric-Stochastic Surrogate

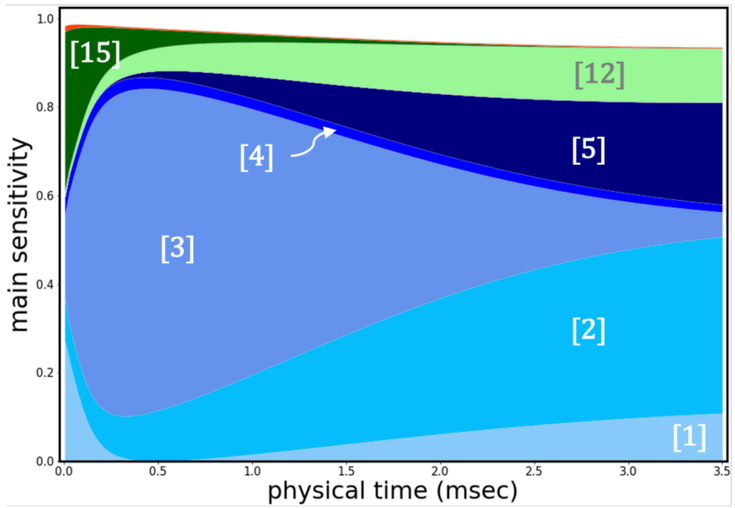
BES Exascale Catalytic Chemistry



Rosenblatt transformation merged with parametric fit to build a joint stochastic-parametric PC surrogate

Karhunen-Loève expansion to extend it to random fields

Global sensitivity analysis of parameteric and stochastic uncertainties wrt time-dependent QoIs



Figs courtesy of Joy Bahr-Mueller

What else

What is in UQTk but not in PyTUQ:

Low-rank tensor (canonical) decomposition

Sparse quadrature

Weighted BCS

PC germ maps

Non-standard, custom PC

Transitional MCMC, Single-site MCMC

Some evidence computation scripts

Data-free inference

Intrusive arithmetics

Coming up (hopefully):

Variational Inference

More optimization routines

More tests needed

More workflows

Math documentation/manual

Usage tutorial of some sort?

Talk at SIAM UQ 26

Part of FASTMath Software catalogue

Any more ideas/questions?