## Contents

1 License 3

2 `pycollocation` 5  
   2.1 `pycollocation` package ..................................................... 5

3 Indices and tables 9

Python Module Index 11
The MIT License (MIT)

Copyright (c) 2015 David R. Pugh

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
2.1 pycollocation package

2.1.1 Submodules

2.1.2 pycollocation.boundary_value_problems module

2.1.3 pycollocation.differential_equations module

2.1.4 pycollocation.models module

2.1.5 pycollocation.orthogonal_polynomials module

2.1.6 pycollocation.solvers module

Objects imported here will live in the pycollocation.solvers namespace

```python
class pycollocation.solvers.LeastSquaresSolver(basis_functions):
    Bases: pycollocation.solvers.solvers.Solver

Attributes

    basis_functions : Functions used to approximate the solution to a boundary value problem.

Methods

    solve(basis_kwargs, boundary_points, ...)
    Solve a boundary value problem using the collocation method.

    solve(basis_kwargs, boundary_points, coefs_array, nodes, problem, **solver_options)
    Solve a boundary value problem using the collocation method.

    Parameters

        basis_kwargs : dict
            Dictionary of keyword arguments used to build basis functions.

        coefs_array : numpy.ndarray
            Array of coefficients for the basis functions.
```
Array of coefficients for basis functions defining the initial condition.

**problem** : bvp.TwoPointBVPLike

A two-point boundary value problem (BVP) to solve.

**solver_options** : dict

Dictionary of options to pass to the non-linear equation solver.

```python
class pycollocation.solvers.Solver(basis_functions)
    Bases: pycollocation.solvers.solvers.SolverLike
```

**Attributes**

| basis_functions | Functions used to approximate the solution to a boundary value problem. |

**Methods**

```python
solve(basis_kwars, boundary_points, ...)  Solve a boundary value problem using the collocation method.
```

```python
solve(basis_kwars, boundary_points, coefs_array, nodes, problem, **solver_options)
    Solve a boundary value problem using the collocation method.
```

**Parameters**

- **basis_kwars** : dict
  Dictionary of keyword arguments used to build basis functions.

- **coefs_array** : numpy.ndarray
  Array of coefficients for basis functions defining the initial condition.

- **problem** : bvp.TwoPointBVPLike
  A two-point boundary value problem (BVP) to solve.

- **solver_options** : dict
  Dictionary of options to pass to the non-linear equation solver.

```python
class pycollocation.solvers.Solution(basis_kwars, functions, nodes, problem, residual_function, result)
    Bases: pycollocation.solvers.solutions.SolutionLike
```

Class representing the solution to a Boundary Value Problem (BVP).

**Attributes**

- **basis_kwars**
- **functions**
- **nodes**
- **problem**
- **residual_function**
- **result**
Methods

```python
evaluate_residual(points)
evaluate_solution(points)
normalize_residuals(points)  # Normalize residuals by the level of the variable.
```

evaluate_residual(points)
evaluate_solution(points)
normalize_residuals(points)  # Normalize residuals by the level of the variable.

```python
class pycollocation.solvers.SolutionLike
    Bases: object

    Attributes
```

```python
    basis_kwargs
    functions
    nodes
    problem
    residual_function
    result
```

class pycollocation.solvers.SolverLike
    Bases: object

    Class describing the protocol the all SolverLike objects should satisfy.

    Notes

    Subclasses should implement solve method as described below.

    Attributes

    ```python
    basis_functions
    ```

    Functions used to approximate the solution to a boundary value problem.

    Methods
solve(basis_kwars, boundary_points, ...) Solve a boundary value problem using the collocation method.

**basis_functions**
Functions used to approximate the solution to a boundary value problem.

- **Getter** Return the current basis functions.
- **Type** basis_functions.BasisFunctions

**solve** (basis_kwars, boundary_points, coefs_array, nodes, problem, **solver_options)
Solve a boundary value problem using the collocation method.

**Parameters**
- **basis_kwars** : dict
  Dictionary of keyword arguments used to build basis functions.
- **coefs_array** : numpy.ndarray
  Array of coefficients for basis functions defining the initial condition.
- **problem** : bvp.TwoPointBVPLike
  A two-point boundary value problem (BVP) to solve.
- **solver_options** : dict
  Dictionary of options to pass to the non-linear equation solver.

2.1.7 pycolloaction.symbolics module
2.1.8 pycolloaction.version module
2.1.9 pycolloaction.visualizers module
2.1.10 Module contents

Objects imported here will live in the pycolloaction namespace
CHAPTER 3

Indices and tables

- genindex
- modindex
- search
p
pycollocation, 8
pycollocation.solvers, 5
B
basis_functions (pycollocation.solvers.SolverLike attribute), 8
basis_kwargs (pycollocation.solvers.SolutionLike attribute), 7

E
evaluate_residual() (pycollocation.solvers.Solution method), 7
evaluate_solution() (pycollocation.solvers.Solution method), 7

F
functions (pycollocation.solvers.SolutionLike attribute), 7

L
LeastSquaresSolver (class in pycollocation.solvers), 5

N
nodes (pycollocation.solvers.SolutionLike attribute), 7
normalize_residuals() (pycollocation.solvers.Solution method), 7

P
problem (pycollocation.solvers.SolutionLike attribute), 7
pycollocation (module), 8
pycollocation.solvers (module), 5

R
residual_function (pycollocation.solvers.SolutionLike attribute), 7
result (pycollocation.solvers.SolutionLike attribute), 7

S
Solution (class in pycollocation.solvers), 6
SolutionLike (class in pycollocation.solvers), 7
solve() (pycollocation.solvers.LeastSquaresSolver method), 5
solve() (pycollocation.solvers.Solver method), 6
solve() (pycollocation.solvers.SolverLike method), 8
Solver (class in pycollocation.solvers), 6
SolverLike (class in pycollocation.solvers), 7