bayesian-filters-smoothers

None

None

None

Table of contents

1. bayesian-filters-smoothers

A unified library for Bayesian filter and smoother algorithms based on the book "Bayesian Filtering and Smoothing" - Simo Sarkka

Purpose of the Package

- To provide a collection of all the Bayesian Filters and Smoothers in a unified framework
- To be a building block for Bayesian Inference of state-space models
- To provide building blocks for state estimation algorithms
- To provide building blocks for parameter estimation algorithms

1.0.1 Features

- Kalman Filter/Smoother
- Extended Kalman Filter/Smoother
- Unscented Kalman Filter/Smoother
- Gaussian Filter/Smoother
- Particle Filter/Smoother

1.0.2 Getting Started

The package can be found on pypi hence you can install it using pip

Installation

```
pip install bayesian-filters-smoothers
```

1.0.3 Documentation

You can check out the full documentation here

1.0.4 Contribution

Contributions are welcome. Notice a bug let us know. Thanks!

1.0.5 Author

- Ninad Kiran Gaikwad, PhD Student, Washington State University

2. API

API Reference

This page gives an overview of all the Bayesian Filters and Smoothers implemented in the package

- Kalman Filter/Smoother
- Extended Kalman Filter/Smoother
- Unscented Kalman Filter/Smoother
- Gaussian Filter/Smoother
- Particle Filter/Smoother

Kalman_Filter_Smoother Class: Kalman Filter/Smoother is the simplest Bayesian Filter/Smoother applied to a linear system with gaussian process/measurement errors

Creates an object to handle Kalman Filtering and Smoothing for Linear Time Invariant (LTI) System

Attributes:

- **A** (`numpy.array`) – LTI system matrix
- **B** (`numpy.array`) – LTI input matrix
- **C** (`numpy.array`) – LTI output matrix
- **m_k** (`numpy.array`) – State mean vector
- **P_k** (`numpy.array`) – State covariance matrix
- **Q** (`numpy.array`) – Process error covariance matrix
- **R** (`numpy.array`) – Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
class Kalman_Filter_Smoother:
"""Creates an object to handle Kalman Filtering and Smoothing for Linear Time Invariant (LTI) System Attributes: A (numpy.ar
"""Initializes the instance of class Kalman_Filter_Smoother Args: A (numpy.array): LTI system matrix B (numpy.array): LTI in
"""Sets the A array for a linear time-varying (LTV) system Args: A (numpy.array): LTV system matrix """ self.A = A def set_B
"""Sets the B matrix for a linear time-varying (LTV) system Args: B (numpy.array): LTV input matrix """ self.B = B def set_C
"""Sets the C matrix for a linear time-varying (LTV) system Args: C (numpy.array): LTV output matrix """ self.C = C def set_
"""Sets the Q matrix for the system Args: Q (numpy.array): Process error covariance matrix """ self.Q = Q def set_R(self, R)
"""Sets the R matrix for the system Args: R (numpy.array): Measurement error covariance matrix """ self.R = R def Kalman_Pre
"""Performs the predict step of the Kalman Filter Args: u_k (numpy.array): Input to the system Returns: m_k_ (numpy.array):
"""Performs the update step of the Kalman Filter Args: y_k (numpy.array): New measurement from the system m_k_ (numpy.array)
"""Performs the smoothening step of the Kalman Smoother Args: u_k_list (list of numpy.array): List of inputs to the system y
```

2.0.1 `Kalman_Predict(u_k)`

Performs the predict step of the Kalman Filter

Parameters:

- **u_k** (`numpy.array`) –

  Input to the system

Returns:

- **m_k_** (`numpy.array`) – Predicted state mean vector
- **P_k_** (`numpy.array`) –

  Predicted state covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
137     def Kalman_Predict(self, u_k):
138 """Performs the predict step of the Kalman Filter Args: u_k (numpy.array): Input to the system Returns: m_k_ (numpy.array):
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
```

2.0.2 `Kalman_Smoother(u_k_list, y_k_list)`

Performs the smoothening step of the Kalman Smoother

Parameters:

- **u_k_list** (`list of numpy.array`) – List of inputs to the system
- **y_k_list** (`list of numpy.array`) – List of measurements from the system

Returns:

- **G_k_list**(`list of numpy.array`) – List of Kalman Smoother Gain
- **m_k_s**(`list of numpy.array`) – List of smoothed state vectors
- **P_k_s**(`list of numpy.array`) – List of smoothed state covarivance matrices

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
189 def Kalman_Smoother(self, u_k_list, y_k_list):
190 """Performs the smoothening step of the Kalman Smoother Args: u_k_list (list of numpy.array): List of inputs to the system y
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
```

```
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
```

2.0.3 `Kalman_Update(y_k, m_k_, P_k_)`

Performs the update step of the Kalman Filter

Parameters:

- **y_k** (`numpy.array`) – New measurement from the system
- **m_k_** (`numpy.array`) – Predicted state mean vector
- **P_k_** (`numpy.array`) – Predicted state covariance matrix

Returns:

- **v_k**(`numpy.array`) – Updated measurement error
- **S_k**(`numpy.array`) – Updated measurement covariance matrix
- **K_k**(`numpy.array`) – Updated Kalman Gain

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
157  def Kalman_Update(self, y_k, m_k_, P_k_):
158  """Performs the update step of the Kalman Filter Args: y_k (numpy.array): New measurement from the system m_k_ (numpy.array)
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
```

2.0.4 `__init__(A, B, C, m_k, P_k, Q, R)`

Initializes the instance of class Kalman_Filter_Smoother

Parameters:

- **A** (`numpy.array`) – LTI system matrix
- **B** (`numpy.array`) – LTI input matrix
- **C** (`numpy.array`) – LTI output matrix
- **m_k** (`numpy.array`) – State mean vector
- **P_k** (`numpy.array`) – State covariance matrix
- **Q** (`numpy.array`) – Process error covariance matrix
- **R** (`numpy.array`) –

  Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
75  def __init__(self, A, B, C, m_k, P_k, Q, R):
76  """Initializes the instance of class Kalman_Filter_Smoother Args: A (numpy.array): LTI system matrix B (numpy.array): LTI inp
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
```

2.0.5 `set_A(A)`

Sets the A array for a linear time-varying (LTV) system

Parameters:

- **A** (`numpy.array`) – LTV system matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
97  def set_A(self, A):
98  """Sets the A array for a linear time-varying (LTV) system Args: A (numpy.array): LTV system matrix """ self.A = A
99
100
101
102
103
```

2.0.6 `set_B(B)`

Sets the B matrix for a linear time-varying (LTV) system

Parameters:

- **B** (`numpy.array`) – LTV input matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
105 def set_B(self, B):
106 """Sets the B matrix for a linear time-varying (LTV) system Args: B (numpy.array): LTV input matrix """ self.B = B
107
108
109
110
111
```

2.0.7 `set_C(C)`

Sets the C matrix for a linear time-varying (LTV) system

Parameters:

- **C** (`numpy.array`) –

LTV output matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
113 def set_C(self, C):
114 """Sets the C matrix for a linear time-varying (LTV) system Args: C (numpy.array): LTV output matrix """ self.C = C
115
116
117
118
119
```

2.0.8 `set_Q(Q)`

Sets the Q matrix for the system

Parameters:

- **Q** (`numpy.array`) –

Process error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
121 def set_Q(self, Q):
122 """Sets the Q matrix for the system Args: Q (numpy.array): Process error covariance matrix """ self.Q = Q
123
124
125
126
127
```

2.0.9 `set_R(R)`

Sets the R matrix for the system

Parameters:

- **R** (`numpy.array`) –

Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
129 
130  def set_R(self, R):
131 """Sets the R matrix for the system Args: R (numpy.array): Measurement error covariance matrix """ self.R = R
132 
133 
134 
135 
```

API Reference

This page gives an overview of all the Bayesian Filters and Smoothers implemented in the package

- Kalman Filter/Smoother
- Extended Kalman Filter/Smoother
- Unscented Kalman Filter/Smoother
- Gaussian Filter/Smoother
- Particle Filter/Smoother

Extended_Kalman_Filter_Smoother Class: Extended Kalman Filter/Smoother is the Bayesian Filter/Smoother applied to a nonlinear system with gaussian process/measurement errors and the gaussian approximation is achieved through linearization of the process and measurement models

Creates an object to handle Extended Kalman Filtering and Smoothing for Linear Time Invariant (LTI) System

Attributes:

- **A** (`numpy.array`) – LTI system matrix
- **B** (`numpy.array`) – LTI input matrix
- **C** (`numpy.array`) – LTI output matrix
- **D** (`numpy.array`) – LTI feedforward matrix
- **m_k** (`numpy.array`) – State mean vector
- **P_k** (`numpy.array`) – State covariance matrix
- **Q** (`numpy.array`) – Process error covariance matrix
- **R** (`numpy.array`) –

  Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
class Extended_Kalman_Filter_Smoother:
"""Creates an object to handle Extended Kalman Filtering and Smoothing for Linear Time Invariant (LTI) System Attributes: A
"""Initializes the instance of class Extended_Kalman_Filter_Smoother Args: f (function(state,input) => numpy.array): Non-lin
"""Sets the system dynamics function for a non-linear time-varying (LTV) system Args: f (function(state,input) => numpy.arra
"""Sets the linearized system dynamics function for a non-linear time-varying (LTV) system Args: F (function(state,input) =>
"""Sets the measurement function for a non-linear time-varying (LTV) system Args: h (function(state) => numpy.array): Non-li
"""Sets the linearized measurement function for a non-linear time-varying (LTV) system Args: H (function(state) => numpy.arr
"""Sets the Q matrix for the system Args: Q (numpy.array): Process error covariance matrix """ self.Q = Q def set_R(self, R)
"""Sets the R matrix for the system Args: R (numpy.array): Measurement error covariance matrix """ self.R = R def Extended_K
"""Performs the predict step of the Extended Kalman Filter Args: u_k (numpy.array): Input to the system Returns: m_k_ (numpy
"""Performs the update step of the Extended Kalman Filter Args: y_k (numpy.array): New measurement from the system m_k_ (num
"""Performs the smoothening step of the Extended Kalman Smoother Args: u_k_list (list of numpy.array): List of inputs to the
```

```
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
```

2.0.10 `Extended_Kalman_Predict(u_k)`

Performs the predict step of the Extended Kalman Filter

Parameters:

- **u_k** (`numpy.array`) – Input to the system

Returns:

- **m_k_**(`numpy.array`) – Predicted state mean vector
- **P_k_**(`numpy.array`) – Predicted state covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
362 def Extended_Kalman_Predict(self, u_k):
363 """Performs the predict step of the Extended Kalman Filter Args: u_k (numpy.array): Input to the system Returns: m_k_ (numpy
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
```

2.0.11 `Extended_Kalman_Smoother(u_k_list, y_k_list)`

Performs the smoothening step of the Extended Kalman Smoother

Parameters:

- **u_k_list** (`list of numpy.array`) – List of inputs to the system
- **y_k_list** (`list of numpy.array`) – List of measurements from the system

Returns:

- **G_k_list**(`list of numpy.array`) – List of Extended Kalman Smoother Gain
- **m_k_s**(`list of numpy.array`) – List of smoothed state vectors
- **P_k_s**(`list of numpy.array`) – List of smoothed state covarivance matrices

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
414 def Extended_Kalman_Smoother(self, u_k_list, y_k_list):
415 """Performs the smoothening step of the Extended Kalman Smoother Args: u_k_list (list of numpy.array): List of inputs to the
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
```

```
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
```

2.0.12 `Extended_Kalman_Update(y_k, m_k_, P_k_)`

Performs the update step of the Extended Kalman Filter

Parameters:

- **y_k** (`numpy.array`) – New measurement from the system
- **m_k_** (`numpy.array`) – Predicted state mean vector
- **P_k_** (`numpy.array`) – Predicted state covariance matrix

Returns:

- **v_k**(`numpy.array`) – Updated measurement error
- **S_k**(`numpy.array`) – Updated measurement covariance matrix
- **K_k**(`numpy.array`) – Updated Extended Kalman Gain

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
382 def Extended_Kalman_Update(self, y_k, m_k_, P_k_):
383 """Performs the update step of the Extended Kalman Filter Args: y_k (numpy.array): New measurement from the system m_k_ (nur
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
```

2.0.13 `__init__(f, F, h, H, m_k, P_k, Q, R)`

Initializes the instance of class Extended_Kalman_Filter_Smoother

Parameters:

- **f** (`function(state,input) => numpy.array`) – Non-linear dynamics function which outputs state vector
- **F** (`function(state,input) => numpy.array`) – Linearized system dynamics which outputs a matrix
- **h** (`function(state) => numpy.array`) – Non-linear output function which outputs mesurement vector
- **H** (`function(state) => numpy.array`) – Linearized output function which outputs a matrix
- **m_k** (`numpy.array`) – State mean vector
- **P_k** (`numpy.array`) – State covariance matrix
- **Q** (`numpy.array`) – Process error covariance matrix
- **R** (`numpy.array`) – Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
290 def __init__(self, f, F, h, H, m_k, P_k, Q, R):
291 """Initializes the instance of class Extended_Kalman_Filter_Smoother Args: f (function(state,input) => numpy.array): Non-lin
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
```

2.0.14 `set_F(F)`

Sets the linearized system dynamics function for a non-linear time-varying (LTV) system

Parameters:

- **F** (`function(state,input) => numpy.array`) – Linearized system dynamics which outputs a matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
322 def set_F(self, F):
323 """Sets the linearized system dynamics function for a non-linear time-varying (LTV) system Args: F (function(state,input) =>
324
325
326
327
328
```

2.0.15 `set_H(H)`

Sets the linearized measurement function for a non-linear time-varying (LTV) system

Parameters:

- **H** (`function(state) => numpy.array`) – Linearized output function which outputs a matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
338 def set_H(self, H):
339 """Sets the linearized measurement function for a non-linear time-varying (LTV) system Args: H (function(state) => numpy.arr
340
341
342
343
344
```

2.0.16 `set_Q(Q)`

Sets the Q matrix for the system

Parameters:

- **Q** (`numpy.array`) – Process error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
346 def set_Q(self, Q):
347 """Sets the Q matrix for the system Args: Q (numpy.array): Process error covariance matrix """ self.Q = Q
348
349
350
351
352
```

2.0.17 `set_R(R)`

Sets the R matrix for the system

Parameters:

- **R** (`numpy.array`) – Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
354 def set_R(self, R):
355 """Sets the R matrix for the system Args: R (numpy.array): Measurement error covariance matrix """ self.R = R
356
357
358
359
360
```

2.0.18 `set_f(f)`

Sets the system dynamics function for a non-linear time-varying (LTV) system

Parameters:

- **f** (`function(state,input) => numpy.array`) – Non-linear dynamics function which outputs state vector

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
314     def set_f(self, f):
315 """Sets the system dynamics function for a non-linear time-varying (LTV) system Args: f (function(state,input) => numpy.arra
316
317
318
319
320
```

2.0.19 `set_h(h)`

Sets the measurement function for a non-linear time-varying (LTV) system

Parameters:

- **h** (`function(state) => numpy.array`) – Non-linear output function which outputs mesurement vector

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
330     def set_h(self, h):
331 """Sets the measurement function for a non-linear time-varying (LTV) system Args: h (function(state) => numpy.array): Non-li
332
333
334
335
336
```

API Reference

This page gives an overview of all the Bayesian Filters and Smoothers implemented in the package

- Kalman Filter/Smoother
- Extended Kalman Filter/Smoother
- Unscented Kalman Filter/Smoother
- Gaussian Filter/Smoother
- Particle Filter/Smoother

Unscented_Kalman_Filter_Smoother Class: Unscented Kalman Filter/Smoother is the Bayesian Filter/ Smoother applied to a nonlinear system with gaussian process/measurement errors and the gaussian approximation is achieved through the unscented transform

Creates an object to handle Unscented Kalman Filtering and Smoothing for Linear Time Invariant (LTI) System

- **f** (`function(state,input) => numpy.array`) – Non-linear dynamics function which outputs state vector
- **h** (`function(state) => numpy.array`) – Non-linear output function which outputs mesurement vector
- **n** (`integer`) – Number of states
- **alpha** (`float`) – Parameter for the UKF algorithm

Attributes:

- **k** (`float`) – Parameter for the UKF algorithm
- **beta** (`float`) – Parameter of the UKF algorithm
- **m_k** (`numpy.array`) – State mean vector
- **P_k** (`numpy.array`) – State covariance matrix
- **Q** (`numpy.array`) – PRocess error covariance matrix
- **R** (`numpy.array`) – Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
class Unscented_Kalman_Filter_Smoother:
"""Creates an object to handle Unscented Kalman Filtering and Smoothing for Linear Time Invariant (LTI) System Attributes: 
"""Initializes the instance of class Extended_Kalman_Filter_Smoother Args: f (function(state,input) => numpy.array): Non-lin
"""Sets the system dynamics function for a non-linear time-varying (LTV) system Args: f (function(state,input) => numpy.arra
"""Sets the measurement function for a non-linear time-varying (LTV) system Args: h (function(state) => numpy.array): Non-li
"""Sets the Q matrix for the system Args: Q (numpy.array): Process error covariance matrix """ self.Q = Q def set_R(self, R)
"""Sets the R matrix for the system Args: R (numpy.array): Measurement error covariance matrix """ self.R = R def __UKF_Sigm
"""Computes Sigma Points for Unscented Kalman Filter Args: m (numpy.array): State mean vector P (np.array): State covariance
"""Computes Sigma Points through the dyanmic model for the Unscented Kalman Filter Args: Sigma_X_Points_list (list of numpy.
"""Computes Sigma Points through the measurement model for the Unscented Kalman Filter Args: Sigma_X_Points_list (list of nu
"""Computes predict step state mean, covariance and cross-coavriance for the Unscented Kalman Filter Args: Sigma_X_Points_li
"""Computes predict step state mean and coavriance for the Unscented Kalman Filter Args: Sigma_X_Points_list (list of numpy.
"""Performs the predict step of the Unscented Kalman Filter Args: u_k (numpy.array): Input to the system Returns: m_k_ (nump
"""Performs the update step of the Unscented Kalman Filter Args: y_k (numpy.array): New measurement from the system m_k_ (nu
"""Performs the smoothening step of the Unscented Kalman Smoother Args: u_k_list (list of numpy.array): List of inputs to th
```

```
975
976
977
978
979
980
981
982
983
984
985
986
987
```

2.0.20 `Unscented_Kalman_Predict(u_k)`

Performs the predict step of the Unscented Kalman Filter

Parameters:

- **u_k** (`numpy.array`) – Input to the system

Returns:

- **m_k_**(`numpy.array`) – Predicted state mean vector
- **P_k_**(`numpy.array`) – Predicted state covariance matrix
- **D_k**(`numpy.array`) – Predicted state cross-covariance

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
836 def Unscented_Kalman_Predict(self, u_k):
837 """Performs the predict step of the Unscented Kalman Filter Args: u_k (numpy.array): Input to the system Returns: m_k_ (nump
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
```

2.0.21 `Unscented_Kalman_Smoother(u_k_list, y_k_list)`

Performs the smoothening step of the Unscented Kalman Smoother

Parameters:

- **u_k_list** (`list of numpy.array`) – List of inputs to the system
- **y_k_list** (`list of numpy.array`) –
  List of measurements from the system

Returns:

- **G_k_list**(`list of numpy.array`) – List of Extended Kalman Smoother Gain
- **m_k_s**(`list of numpy.array`) – List of smoothed state vectors
- **P_k_s**(`list of numpy.array`) –
  List of smoothed state covarivance matrices

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
def Unscented_Kalman_Smoother(self, u_k_list, y_k_list):
"""Performs the smoothening step of the Unscented Kalman Smoother Args: u_k_list (list of numpy.array): List of inputs to th
```

2.0.22 `Unscented_Kalman_Update(y_k, m_k_, P_k_)`

Performs the update step of the Unscented Kalman Filter

Parameters:

- **y_k** (`numpy.array`) – New measurement from the system
- **m_k_** (`numpy.array`) – Predicted state mean vector
- **P_k_** (`numpy.array`) – Predicted state covariance matrix

Returns:

- **mu_k**(`numpy.array`) – Predicted measurement mean
- **S_k**(`numpy.array`) – Predicted measurement covariance matrix
- **C_k**(`numpy.array`) – Predicted state-measurement covariance matrix
- **K_k**(`numpy.array`) – Updated Extended Kalman Gain

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
860 def Unscented_Kalman_Update(self, y_k, m_k_, P_k_):
861 """Performs the update step of the Unscented Kalman Filter Args: y_k (numpy.array): New measurement from the system m_k_ (nu
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
```

2.0.23 __UKF_Predict_State_MeanCovariance(Sigma_X_Points_list, Sigma_X_Points_Tilde_list)

Computes predict step state mean, covariance and cross-coavriance for the Unscented Kalman Filter

Parameters:

- **Sigma_X_Points_list** (`list of numpy.array`) – List of Sigma points
- **Sigma_X_Points_Tilde_list** (`(list of numpy.array`) – List of Sigma points passed through the dynamic model

Returns:

- **m_k_**(`numpy.array`) – Predicted state mean vector
- **P_k_**(`numpy.array`) – Predicted state covariance matrix
- **D_k**(`numpy.array`) – Predicted state cross-covariance

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
691 def __UKF_Predict_State_MeanCovariance(self, Sigma_X_Points_list, Sigma_X_Points_Tilde_list):
692 """Computes predict step state mean, covariance and cross-coavriance for the Unscented Kalman Filter Args: Sigma_X_Points_li
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
```

```
750
751
752
753
754
755
756
757
758
759
760
761
```

2.0.24 __UKF_SigmPoints_DynamicModel(Sigma_X_Points_list, u_k)

Computes Sigma Points through the dyanmic model for the Unscented Kalman Filter

Parameters:

- **Sigma_X_Points_list** (`list of numpy.array`) – List of Sigma points
- **u_k** (`numpy.array`) – Input to the system

Returns:

- **Sigma_X_Points_Tilde_list**(`list of numpy.array`) – List of Sigma points passed through the dynamic model

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
642 def __UKF_SigmPoints_DynamicModel(self, Sigma_X_Points_list, u_k):
643 """Computes Sigma Points through the dyanmic model for the Unscented Kalman Filter Args: Sigma_X_Points_list (list of numpy.
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
```

2.0.25 `__UKF_SigmPoints_Generator(m, P)`

Computes Sigma Points for Unscented Kalman Filter

Parameters:

- **m** (`numpy.array`) – State mean vector
- **P** (`np.array`) –
  State covariance matrix

Returns:

- **Sigma_X_Points_list(** `list of numpy.array` **)** –
  List of Sigma points

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
587 def __UKF_SigmPoints_Generator(self, m, P):
588 """Computes Sigma Points for Unscented Kalman Filter Args: m (numpy.array): State mean vector P (np.array): State covariance
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
```

2.0.26 __UKF_SigmPoints_MeasurementModel(Sigma_X_Points_list)

Computes Sigma Points through the measurement model for the Unscented Kalman Filter

Parameters:

- **Sigma_X_Points_list** (list of numpy.array) – List of Sigma points

Returns:

- **Sigma_Y_Points_Tilde_list**(list of numpy.array) – List of Sigma points passed through the measurement model

Source code in bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py

```
667 def __UKF_SigmPoints_MeasurementModel(self, Sigma_X_Points_list):
668 """Computes Sigma Points through the measurement model for the Unscented Kalman Filter Args: Sigma_X_Points_list (list of nu
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
```

2.0.27

`__UKF_Update_StateMeasurement_MeanCovariance(Sigma_X_Points_list, Sigma_Y_Points_Tilde_list, m_k_)`

Computes predict step state mean and coavriance for the Unscented Kalman Filter

Parameters:

- **Sigma_X_Points_list** (`list of numpy.array`) – List of Sigma points
- **Sigma_Y_Points_Tilde_list** (`(list of numpy.array`) – List of Sigma points passed through the measurement model
- **m_k_** (`numpy.array`) – Predicted state mean vector

Returns:

- **mu_k(** `numpy.array` **)** – Predicted measurement mean
- **S_k(** `numpy.array` **)** – Predicted measurement covariance matrix
- **C_k(** `numpy.array` **)** – Predicted state-measurement covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
763 def __UKF_Update_StateMeasurement_MeanCovariance(self, Sigma_X_Points_list, Sigma_Y_Points_Tilde_list, m_k_):
764 """Computes predict step state mean and coavriance for the Unscented Kalman Filter Args: Sigma_X_Points_list (list of numpy
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
```

```
822
823
824
825
826
827
828
829
830
831
832
833
834
```

2.0.28 `__init__(f, h, n, m, alpha, k, beta, m_k, P_k, Q, R)`

Initializes the instance of class Extended_Kalman_Filter_Smoother

Parameters:

- **f** (`function(state,input) => numpy.array`) – Non-linear dynamics function which outputs state vector
- **h** (`function(state) => numpy.array`) – Non-linear output function which outputs mesurement vector
- **n** (`integer`) – Number of states
- **m** (`integer`) – Number of outputs
- **alpha** (`float`) – Parameter for the UKF algorithm
- **k** (`float`) – Parameter for the UKF algorithm
- **m_k** (`numpy.array`) – State mean vector
- **P_k** (`numpy.array`) – State covariance matrix
- **Q** (`numpy.array`) – PRocess error covariance matrix
- **R** (`numpy.array`) – Measurement error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
519 def __init__(self, f, h, n, m, alpha, k, beta, m_k, P_k, Q, R):
520 """Initializes the instance of class Extended_Kalman_Filter_Smoother Args: f (function(state,input) => numpy.array): Non-lin
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
```

2.0.29 `set_Q(Q)`

Sets the Q matrix for the system

Parameters:

- **Q** (`numpy.array`) –

  Process error covariance matrix

Source code in `bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py`

```
571 def set_Q(self, Q):
572 """Sets the Q matrix for the system Args: Q (numpy.array): Process error covariance matrix """ self.Q = Q
573
574
575
576
577
```

2.0.30 set_R(R)

Sets the R matrix for the system

Parameters:
- **R** (numpy.array) – Measurement error covariance matrix

Source code in bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py

```
579 def set_R(self, R):
580 """Sets the R matrix for the system Args: R (numpy.array): Measurement error covariance matrix """ self.R = R
581
582
583
584
585
```

2.0.31 set_f(f)

Sets the system dynamics function for a non-linear time-varying (LTV) system

Parameters:
- **f** (function(state,input) => numpy.array) – Non-linear dynamics function which outputs state vector

Source code in bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py

```
555 def set_f(self, f):
556 """Sets the system dynamics function for a non-linear time-varying (LTV) system Args: f (function(state,input) => numpy.arra
557
558
559
560
561
```

2.0.32 set_h(h)

Sets the measurement function for a non-linear time-varying (LTV) system

Parameters:
- **h** (function(state) => numpy.array) – Non-linear output function which outputs mesurement vector

Source code in bayesian-filters-smoothers\bayesian_filters_smoothers\bayesian_filters_smoothers.py

```
563 def set_h(self, h):
564 """Sets the measurement function for a non-linear time-varying (LTV) system Args: h (function(state) => numpy.array): Non-li
565
566
567
568
569
```

API Reference

This page gives an overview of all the Bayesian Filters and Smoothers implemented in the package

- Kalman Filter/Smoother
- Extended Kalman Filter/Smoother
- Unscented Kalman Filter/Smoother
- Gaussian Filter/Smoother
- Particle Filter/Smoother

API Reference

This page gives an overview of all the Bayesian Filters and Smoothers implemented in the package

- Kalman Filter/Smoother
- Extended Kalman Filter/Smoother
- Unscented Kalman Filter/Smoother
- Gaussian Filter/Smoother
- Particle Filter/Smoother

2.0.33 User Guide

Examples

```
>>> from bayesian_filters_smoothers import addnum >>> addnum(10,5)
```

Jupyter Notebook Examples

Coming soon...