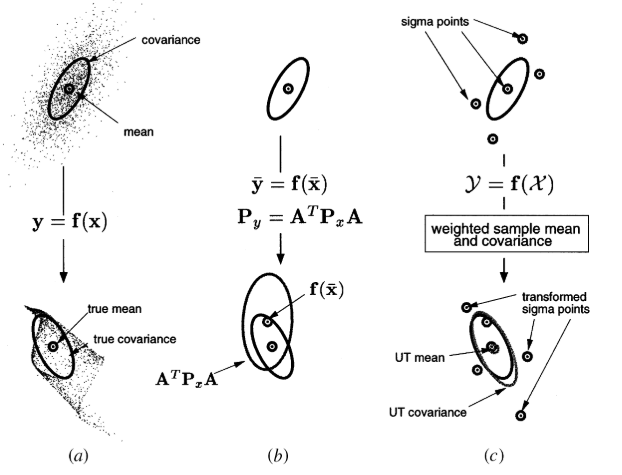
UNSCENTED KALMAN FILTER

# BAB 3 dari paper backup (**sudah sama**)

UKF merupakan salah satu metode pengembangan kalman filter yang bertujuan untuk mengupdate bobot. Metode yang digunakan mirip dengan tipe metode Monte Carlo, dimana sejumlah *sample* dipilih secara acak yang juga diperbanyak melalui transformasi *nonlinear.* Di sisi lain, *sigma points* tidak dipilih secara acak, melainkan secara deterministik dan jumlah *sigma points* yang rendah. Ilustrasi perbedaan UKF dan pengembangan kalman filter linear dan *Extended Kalman Filter* dapat dilihat pada Gambar 3.8.



**Gambar 3.8** Perbandingan a)Nilai sebenarnya, b)EKF, c)UKF

Himpunan *sigma points* dipilih dan digunakan sebagai *sample* untuk *mean* dan kovarian.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | (3.14) | |
|  |  | | (3.15) | |
|  |  | | (3.16) | |
| Keterangan: | | | |  |
| :titik-titik *Sigma* | | :ukuran dimensi data | | |
| : state k-1 | |  | | |

:ukuran dimensi datasvc

ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssscx

|  |  |  |
| --- | --- | --- |
|  |  | (3.17) |
|  |  | (3.18) |
|  |  | (3.19) |
|  |  | (3.20) |

Kemudian vektor sigma diperbanyak melalui fungsi nonlinear

|  |  |  |
| --- | --- | --- |
|  |  | (3.21) |
|  |  |  |
|  |  | (3.22) |

# BAB 3 UTAMA dari internet acuan (MEDIUM) (**SUDAH SAMA)**

Algoritma pembelajaran yang digunakan pada penelitian ini merupakan algoritma gabungan dari *Recurrent Neural Network* dan update bobot *Unscented Kalman Filter*. Berdasarkan persamaan matematis (Wan & Merwe, 2002), *Unscented Kalman Filter* memiliki persamaan dan tahapan seperti seperti berikut:

1. Hitung kumpulan titik sigma
2. Tetapkan bobot untuk setiap titik sigma
3. Transformasikan titik tersebut melalui fungsi non linier
4. Hitung gaussian dari titik yang dibobot (weighted) dan di transformasi
5. Hitung mean dan varian dari Gaussian baru

**Langkah Prediksi**

Maka terdapat langkah prediksi yang mirip dengan 5 langkah tadi, yaitu:

1. Hitung titik sigma (sama dengan langkah 1)
2. Hitung bobot titik sigma (sama dengan langkah 2)
3. Mengubah titik sigma dan hitung mean dan kovariana barudengan pertimbangan *noise* dari proses (**R**)

|  |  |
| --- | --- |
|  | (4.38) |
|  |  |

**Langkah Update**

Setelah memiliki *mean* dan kovarian prediksi, maka dihitung perbedaan antar nilai *mean* dan kovarian yang diprediksi dengan nilai *mean* sebenarnya dan kovarian. Perhitungan ini memiliki prosedur yang mirip dengan persamaan di Kalman Filter. Jadi yang dilakukan selanjutnya adalah mengambil *predicted state* ke *measurement state.*

Disini, terdapat pilihan dalam menghasilkan titik sigma lagi. Pilihan pertama adalah menentukan titik sigma dari awal lagi karena prediksi *mean* dan varian berubah, karena sing mempengaruhi satu sama lain. Pilihan lainnya adalah dengan menggunakan titik sigma yang telah dihasilkan saja. Untuk menggunakan titik sigma yang sebelumnya, maka diambil *state* dari *state space* ke *measurement state space* seperti pada pers.(.

|  |  |
| --- | --- |
|  | (4.43) |
|  | (4.44) |
|  | (4.45) |
| Keterangan: |  |
| :titik sigma yang ditransformasi ke dalam *measurement space* | |
| : *mean* di *measurement* *space* | |
| : kovariansi di *measurement* *space* | |
| : *noise* | |
| : fungsi yang memetakan titik sigma yang ditetapkan sebelumnya ke *measurement* *space* | |

Penting: merupakan *measurement space,* yaitu pengukuranyang berasal dari sensor (sistem RNN). Jadi fungsi bertugas untuk mengubah *state space* menjadi *measurement space* sehingga dapat disamakan ke dalam satu unit.

Disini tidak perlu dihitung nilai Jacobian seperti pada Kalman Filter karena disini tidak akan dilakukan linierisasi fungsi (membuat garis fungsi). Untuk menghitung *Kalman gain*, perlu dilakukan perhitungan korelasi silang antara titik sigma di *state space* dan titik sigma di *measurement space.*

|  |  |
| --- | --- |
|  | (4.46) |
|  | (4.47) |
| Keterangan: |  |
| :korelasi silang matriks antara *state space* dan *predicted space* | |
| : Kalman gain | |
| : Matriks kovarian yang diprediksi | |

**Final state**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | () | |
|  | | () | |
| Keterangan: | |  | |
| : *mean measurement* sebenarnya yang berasal dari sensor | | |  |
| : *mean* di *measurement space* | | |  |
| : *mean* | :prediksi *mean* | | |
| : kovarian | : prediksi kovarian | | |
| : *Kalman gain* | :dimensi | | |
| : *sama* dengandi Kalman Filter dan di EKF | | | |

Notasi merupakan *weight* untuk *mean*

Yang pada tahap ini, pseudocode darrii UKF dapat dirumuskan menjadi:

MULAI :

1. **function** hitungSigma()

**return** *jbcjsbcjsbccmb*(**)**

1. **function** predict()

**return** sxs(**)**

1. **function** update()

**return** predict(**)**

1. **function** UKF()
2. *Zh* ← σ(*wihUi* + *wchCc*)
3. *Yo* ← σ(*whoZh*) SELESAI

Xax

xX

x

MULAI :

1. **function** hitungSigma()

**return** *jbcjsbcjsbccmb*(**)**

1. **function** predict()

**return** sxs(**)**

1. **function** update()

**return** predict(**)**

1. **function** UKF()
2. *Zh* ← σ(*wihUi* + *wchCc*)
3. *Yo* ← σ(*whoZh*) SELESAI

Dimana menggukankan sebagiannnnnn

# **BAB 4 dari hasil backup paper**

Yang tidak dipakai

|  |  |  |
| --- | --- | --- |
| **Inisialisasi dan transformasi** | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
| For seaharusnyaw ada penjelasannya lagi Tambahin blab la bla | |  |
| **Menghitung titik sigma:** | |  |
|  | | () |
| Keterangan: | |  |
| : matriks transisi | : *noise* dari *process* | |
| : Kovarian *noise* dari *process* | : *state*dari t-1 sampai *t.* | |

Pada pers.(**Error! No text of specified style in document.**.39, menghitung sigma points dengan sebagai kolom ke-i dari matriks akar kuadrat dari dengan digunakan untuk mengontrol kovarian matriks dengan .

|  |  |
| --- | --- |
| Tambahin blab la bla |  |
| ***Time Update:*** |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

axsaxaxa

|  |  |
| --- | --- |
| Tambahin blab la bla |  |
| ***Measurement Update:*** |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Paramerer scalar dari dan menentukan penyebaran dari *sigma points* di sekitar . Dimana juga mempengaruhi pengurangan order error dari pendekatan mean dan kovarian. Konstan biasanya ditetapkan antara [-4,1], dan atau 0 (Bisoi & Dash, 2014).

|  |  |
| --- | --- |
| Tambahin blab la bla |  |
|  |  |
|  |  |
| dengan dan |  |
|  |  |
|  |  |
|  |  |
| Dengan *priori* dari error kovarian dijabarkan dengan |  |
|  |  |
|  |  |
|  |  |
|  |  |

*Sigma point* kemudian diperbanyak melallui *output* model prediksi data historis untuk mengestimasi *mean* dan kovarian dari

|  |  |
| --- | --- |
|  |  |
|  |  |
| Step2 : Measurement update???????????????? |  |
|  |  |

* 1. Prosedur Pelatihan

Pada penelitian, dilakukan langkah-langkah yang terdiri dari beberapa tahap, yakni:

* + 1. Data latih

Persentase pemisahan data yang digunakan adalah 70% data dari data keseluruhan yang telah dinormalisasi. Data yang terhitung adalah data sebanyak 1647 data.

* + 1. *Preprocessing*

Dilakukan normalisasi agar jangkauan nilai harga Bitcoin tidak terlalu jauh. Normalisasi dilakukan agar hasil prediksi yang diharapkan dapat maksimal. Normalisasi data akan menghasilkan nilai [-1,1] yang kemudian akan dilakukan pemilihan data dengan fungsi aktivasi hyperbolic tangent. Penghitungan dilakukan pada data dengan jumlah input yang telah ditentukan dengan metode *sliding window*.

* + 1. Inisialisasi bobot awal

Terdapat tiga bobot awal yang ditentukan secara random [-1,1] untuk digunakan dalam aristektur RNN yaitu sebagai betikut:

1. sebagai bobot dari *input* ke *context layer.*

2. sebagai bobot dari *context* ke *hidden layer.*

3. sebagai bobot dari *hidden* ke *output layer.*

* + 1. Inisialisasi parameter pelatihan

Parameter pembelajaran ditentukan pada awal sistem. Dalam sistem ini, parameter pembelajarannya adalah sebagai berikut:

Tabel **Error! No text of specified style in document.**.3 Parameter pelatihan

|  |  |  |
| --- | --- | --- |
| No. | Parameter | Penjelasan |
| 1. | *Process noise covariance Q* | Matriks berukuran , sebagai kuantitas dari bobot. |
| 2. | *Measurement noise covariance R* | Matriks berukuran , sebagai jumlah neuron pada *output layer* |
| 3. | *Error covariance P* | Matriks berukuran , diinisialisasikan dengan nilai |
| 4. | Jumlah *epoch* | Diterapkan sebanyak 2000 epoch |
| 5. | Fungsi aktivasi | Fungsi aktivasi linear dan *hyperbolic tangent* |

*Q* dan *R* memiliki nilai dimana merupakan skalar dan merupakan matriks identitas. Akan dilakukan pengujian terhadap nilai untuk mengetahui pengaruh nilai *Q* dan *R* terhadap hasil prediksi.