1 Kalman filter Upgrade

1.1 Defining the system

In This project I aim to plot the convergence of the Kalman filter described below;

$$x_{t+1} = Ax_t + w_t, \qquad y_t = Cx_t + v_t,$$

where

$$\tilde{m}_t = A\tilde{m}_{t-1} + \Sigma_{t|t-1}C^T(C\Sigma_{t|t-1}C^T + V)^{-1}(y_t - CA\tilde{m}_{t-1}), \qquad \tilde{m}_t = E[x_t|y_{[0,1]}]$$

$$E[w_t w_t^T] =: W \ \textit{with} \ w_t \sim \mathcal{N}(0, I) \quad \textit{and} \quad E[v_t v_t^T] =: V \ \textit{with} \ v_t \sim \mathcal{N}(0, 1)$$

associated Riccati recursions for the covariance matrix updates

$$\Sigma_{t+1|t} = A\Sigma_{t|t-1}A^T + W - (A\Sigma_{t|t-1}C^T)(C\Sigma_{t|t-1}C^T + V)^{-1}(C\Sigma_{t|t-1}A^T)$$

Implementation of these iterations into Python is as follows:

```
    v_t = np.random.normal(0, 1, 1)
    w_t = np.random.normal(0, 1, (np.size(A, 0), 1))
    y = C @ x + v_t
    x = A @ x + w_t
    M = A @ M + sigma @ C_T * ((y - C @ A @ M) / (C @ sigma @ C_T + 1))
    sigma = A @ sigma @ A.T + I - A @ sigma @ C_T * ((C * sigma @ A.T) /
    (C @ sigma @ C_T + 1))
```

I used matplot library for plotting the system and numpy, these can be installed by typing the following commands to the terminal one by one:

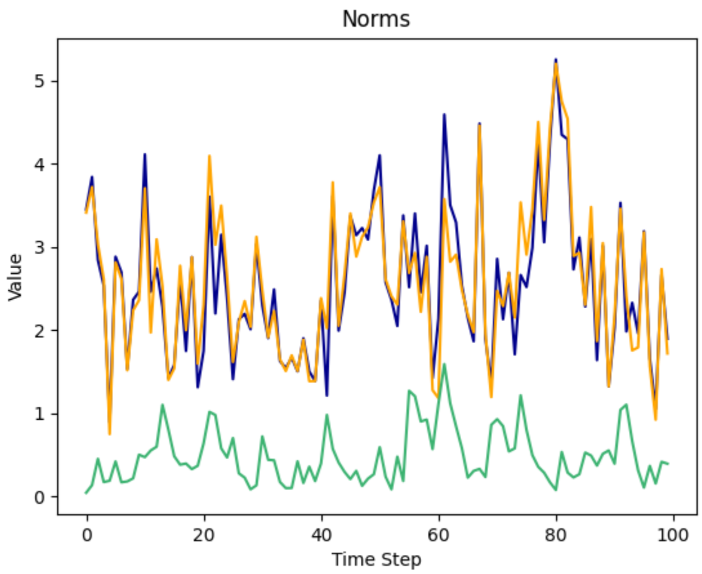
```
pip install matplotlib
pip install numpy
```

The matrices A and C used in the code are just examples; they can be replaced with any $N \times N$ and $N \times 1$ matrices, respectively. For convergence, the eigenvalues of A must be less than 1.

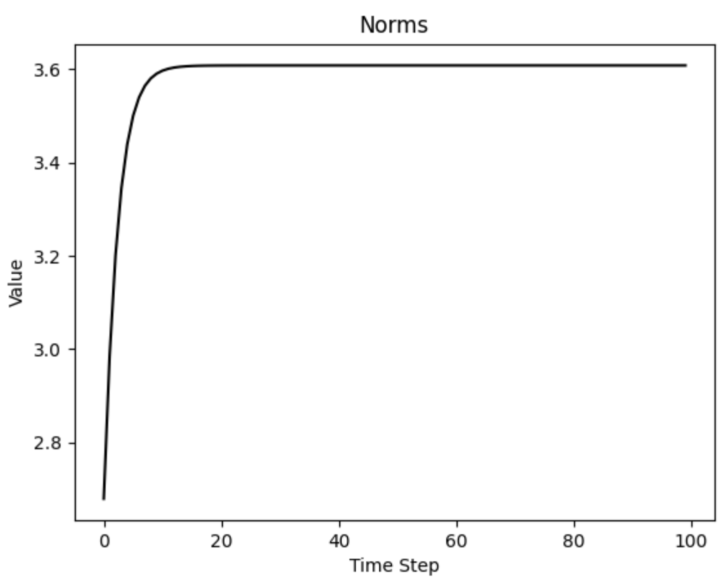
for the following matrices, plots of $|x|$, $|\tilde{m}|$ and $|x - \tilde{m}|$ are as follows

```
A = np.array([[0.8, 0, 0, 0 , 0.2],
              [0, 0.1, 0.1, 0, 0],
              [0, 0, 0.3, 0, 0.1],
              [0, 0, 0, 0.1, 0],
              [0.1 ,0.2 ,0 ,0 ,0 ]])

C = np.array([0.1, 0, 0, 0, 0.2])
```



Here $|x|$ is yellow, $|\tilde{m}|$ is green and $|x - \tilde{m}|$ is blue. Finally, the Frobenius norm of sigma showes the convergence of the system:



Şevket Kaan Alkır $-$ *July***2024**