### Instructions:

When producing motor commands, errors occur due to a wide array of disturbances such as fatigue, damage, or development. All of these disturbances occur at different timescales, and require different adaptations. Problems at a shorter timescale should be forgotten, while longer ones should be kept around. We will apply a Kalman filter to understand this credit assignment problem - given motor error what disturbances are responsible, and how should an optimal agent adapt.

This tutorial requires: sample\_gaussian.m, test\_kalman.m, KTS.m, kalman\_filter.m, gaussian\_prob.m, fitLinear.m, kalman\_update.m, sample\_lds.m, and fitExponential.m (so all the .m files in the folder).

**Step 1:** Read through the paper Kording, Tenembaum, and Shadmehr 2007, to get an understanding of what you are doing. Then download the code to replicate the eye movement simulation figures, specifically figures 2b,c and 3c,g,h. The code here has been edited from the original with some added comments and some key sections removed.

**Step 2:** Fix the kalman\_update.m with the Kalman [update equations](https://en.wikipedia.org/wiki/Kalman_filter#Details). Run the simple example from the test\_kalman.m file with your now working Kalman filter to see how it works.

**Step 3**: Define the parameters for the simulations. Go to the KTS.m file and define the transition and observation matrices from the paper (which are the same parameters for all experimental conditions and figures). Go through and run the code to generate figure 2b once those are defined.

**Step 4:** Once you understand how figure 2b is generated, move on to filling in the code to simulate the experiments for the other figures. Each of these are similar in structure, but require slightly different solutions.

**Step more:** If finished with that, play with the basic model and see how it changes the results. For example, what happens if you have far fewer memory states? Try writing up new code to replicate the contrast adaptation and word learning examples.

**Note on notation**

Apologies about everything ever - there is no standard notation for the Kalman Filter so there are translation issues that for those learning can be a huge wall. For my own sanity, I've made the (hopefully correct) table below to translate the important matrices from the Kalman update [wikipedia page](https://en.wikipedia.org/wiki/Kalman_filter), the paper (Kording, Tenembaum, & Shadmehr 2007), and the accompanying code. Check before trusting this completely.

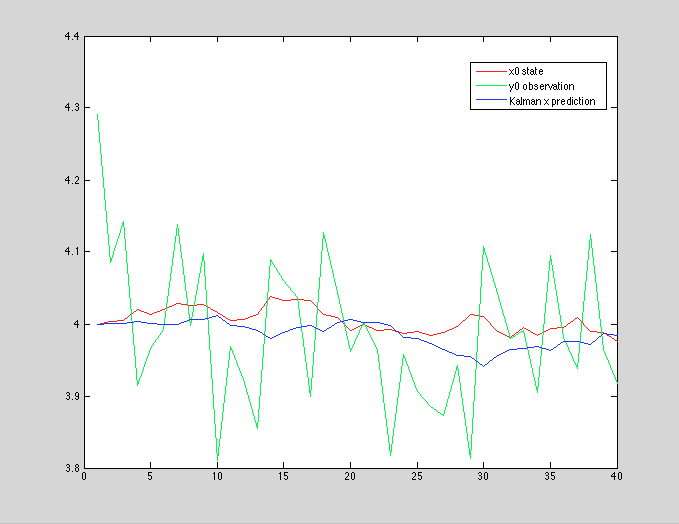
|  |  |  |  |
| --- | --- | --- | --- |
| **Term** | **Wiki** | **Paper** | **Code** |
| State transition model | F | M | A |
| State transition variance | Q | Q | Q |
| Observation model | H | H | C |
| Observation uncertainty | P | V | V |

#### Figures:

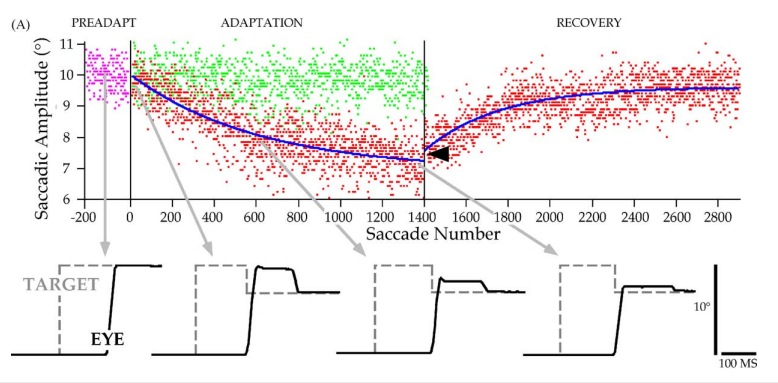
While you should read through the paper, included here are the figures that you will end up producing (in MATLAB). The figure titles here (e.g. 2b) are in reference to the Kording, Tenembaum, and Shadmehr 2007 paper.

**Test\_kalman example:**

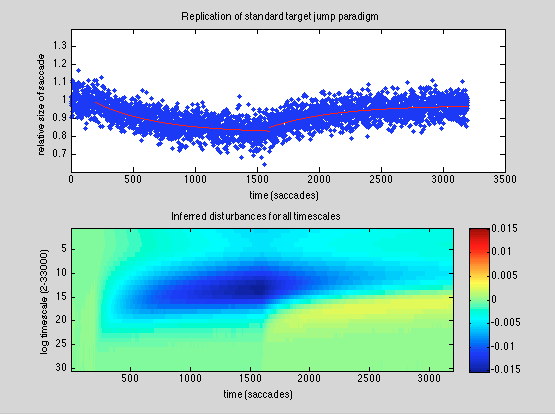
Example using sample\_lds and kalman\_filter, with a constant state transition model (with noise). This is simply one example trace - yours will likely look different.



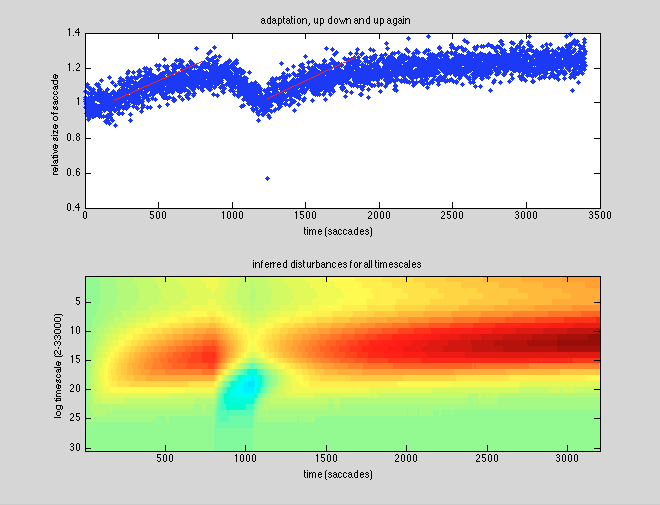
**Figure 2b:**

Originally from [Hopps and Fuchs 2004](https://www.researchgate.net/profile/Albert_Fuchs2/publication/5332275_Hopp_J.J.__Fuchs_A.F._The_characteristics_and_neuronal_substrate_of_saccadic_eye_movement_plasticity._Prog._Neurobiol._72_27-53/links/02e7e533a52bfb9945000000.pdf). In these experiments, subjects had to make a 10 degree saccade (to a particular target), but then (after 200 ‘practice’ trials) the target moves during the saccade. The subject eventually adapts by reducing the gain on the saccade (for an overshot). Then at trial 1400, the target stops moving, and the subject adapts back to the original location.

Here is a figure from Hopps and Fuchs 2004, showing the target and eye positions with respect to the trial.

And here is a figure generated by the MATLAB Kalman filter simulation.

**Figure 3c:**  
Originally from [Kojima et al 2004](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.119.9968&rep=rep1&type=pdf). Double adaptation experiments. Following an initial set of trials, there is positive perturbation of the target by 35% for 800 trials, followed by a 35% negative perturbation of the target (from the initial position). This continues until the gain is back to neutral (i.e. the subject correctly saccades to the initial position), at which time the target is (again) positively perturbed by 35%. This second positive perturbation is followed by a quicker adaptation towards the new target position, indicating that some memory remains of the previous positive perturbation.



#### 

#### 

#### 

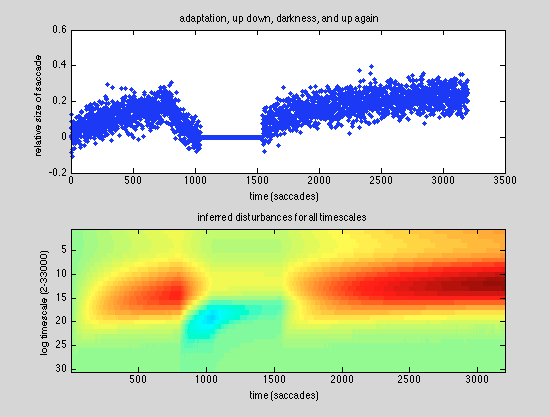
#### 

#### 

#### 

#### **Figure 3g**

Originally from [Kojima et al 2004](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.119.9968&rep=rep1&type=pdf). Here we have a period with no information - after the gain resets following a reversal, the subject is blinded (so no information) and then a positive perturbation is produced. Note that the subject 'lost' some of the recent negative adaptation and showed spontaneous recovery.



#### 

#### 

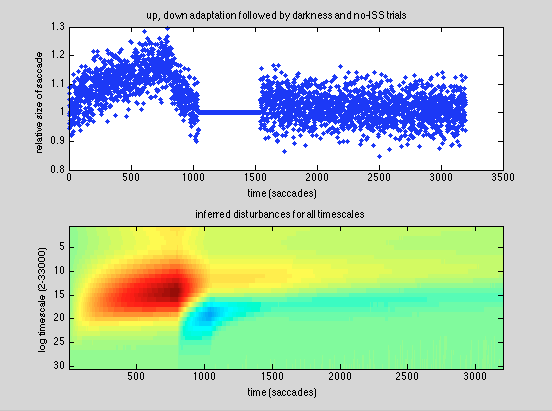
#### 

#### 

#### 

#### **Figure 3h**

Originally from [Kojima et al 2004](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.119.9968&rep=rep1&type=pdf). This experiment is the same as the previous, but instead of a positive perturbation after the dark period, there is no perturbation (so the perturbation is set back to 0). Note here that there is no spontaneous recovery.



#### 

#### 

#### 

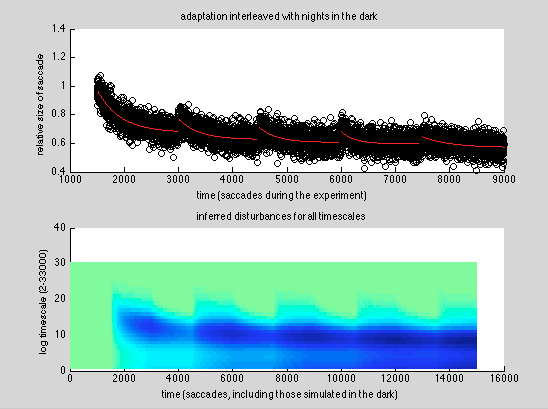
#### 

#### 

#### 

#### **Figure 2d**

Originally from [Robinson et al 2006](http://jn.physiology.org/content/96/3/1030.short). In these experiments, subjects went through the adaptation training (with an offset of 50%) over multiple days, each day having 1500 trials and being blindfolded for the rest of the time. Note that on each day subject's rates of learning is faster, till finally they almost achieve instant adaptation.



**Notes**:

In the kalman\_update.m file I have removed the control component, u. This is important for the full Kalman Filter, but not for our use here so can be ignored (helps clean up the code). However it's very easy to add it back on. There are some other bits that are stripped down or hopefully simplified.

Since this is modified from the original code linked above, you could download the original and look at the solutions to compare your code.