Investigation

Let's make a fun little investigation. Assume we have some data of a signal and we want to make an algorithm that tracks the 1-d signal. This presentation is a little bit too short to make a real algorithm and data in so we will fake it instead.

Synthesize the data

Let's use some Emacs lisp to generate the ground truth and the estimates for us:

```
  (mapcar (lambda (i)
            (list i (+ (random 4) (- i 2))))
          (number-sequence 1 20))

(cons '("Truth" "Estimate") (cons 'hline tbl))
```

Let's give the results a name so that we can reference the table later. It's good to see the data in the table but I often find that some kind of visualization is more powerful. It's too bad I don't know how to plot in elisp... but I do know how to do it in Python.

Visualize estimate and truth

```
import numpy as np
import matplotlib
matplotlib.use('Agg')
from matplotlib import pyplot as plt

# Convert list to numpy array
truth = np.asarray(data)[:, 0]
est = np.asarray(data)[:, 1]
# Plot
fig=plt.figure()
plt.plot(truth, color="g", label="Truth")
plt.plot(est, marker="x", label="Estimate")
plt.legend(loc='upper left')
plt.xlabel("Sample")
plt.ylabel("Value")
plt.title("Tracking")
```

```
plt.savefig('.images/est_vs_truth.png')
'.images/est_vs_truth.png' # return this to org-mode
```

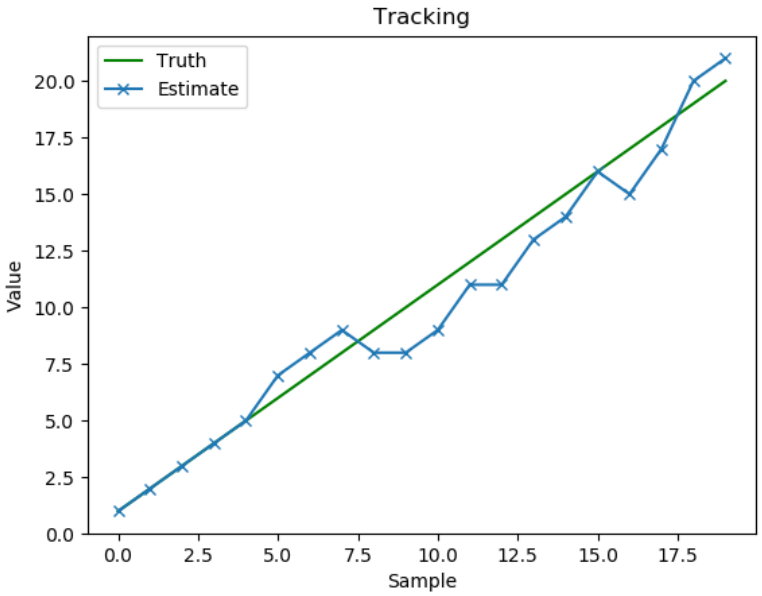

Figure 1: Tracking the true value

Cool, the performance of the fake algorithm is not that bad. I think we can be pretty happy with it. let's see if we can gather some more information about it's performance.

Examine the performance

Truth	Estimate	Error	Absolute error
1	1	0	0
2	2	0	0
3	3	0	0
4	4	0	0
5	5	0	0
6	7	-1	1
7	8	-1	1
8	9	-1	1
9	8	1	1
10	8	2	2
11	9	2	2
12	11	1	1
13	11	2	2
14	13	1	1
15	14	1	1
16	16	0	0
17	15	2	2
18	17	1	1
19	20	-1	1
20	21	-1	1
RMSE	1.140175425099138		

In order to get the values from the other table I am using remote references. To refer to the values of the other table.

Table formulas uses `Emacs calc` syntax by default. I have a hard time not thinking of `Windows calc` when I hear this name. But this is something different.

Tables also supports `Emacs lisp` so we can use that to calculate the absolute error. Finally it would be nice to get a kpi like `rmse` to have one number for the performance of the algorithm. Since that equation would be quite long in Emacs lisp maybe it's time to try something else.

We can actually pass the data from the table into other code blocks, which is a supercool. We can therefore create a block with Python code which we pass values into in order to be able to calculate the rmse

A second try

Describe the flow

I would like to describe the flow better. It would be great if we could visualize it, perhaps in a flow chart.

Export

Improve

Appendix

I feel that the Emacs calculator requires some more data