1. Moving averages [10 pts]

by Roumen Guha on Sunday, February 26th, 2017

There are many ways to model the relationship between an input sequence $\{u_1, u_2, \ldots\}$ and an output sequence $\{y_1, y_2, \ldots\}$. In class, we saw the *moving average* (MA) model, where each output is approximated by a linear combination of the k most recent inputs:

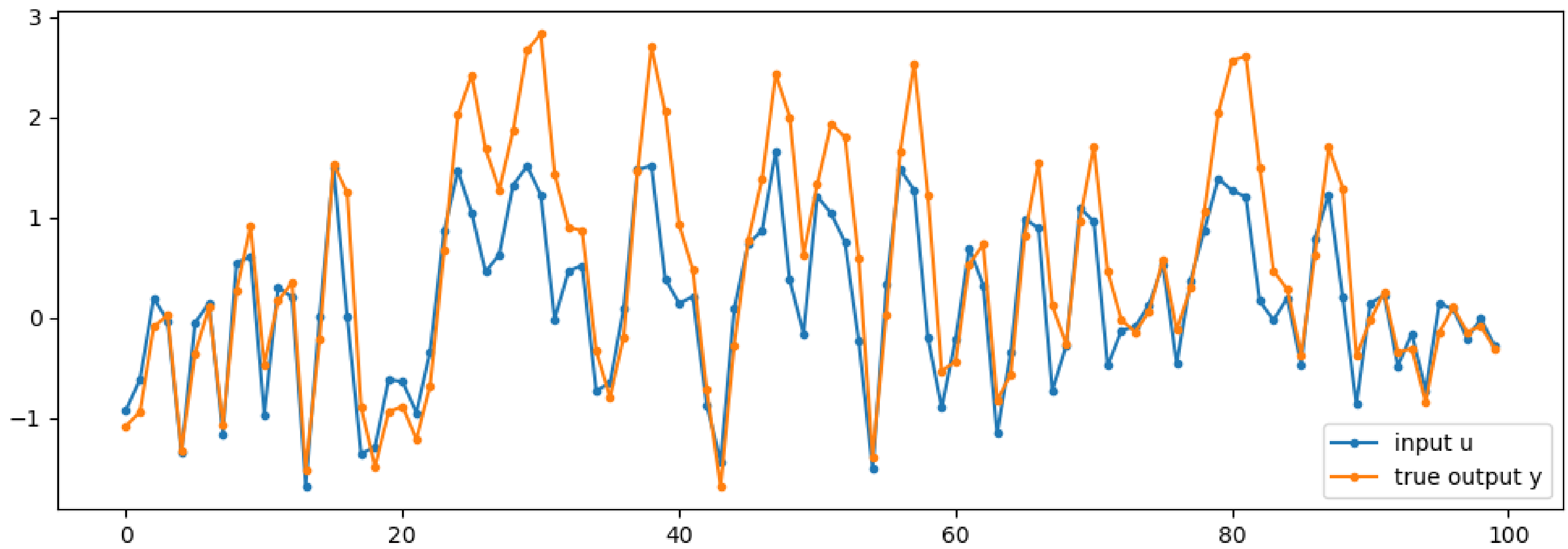
$$\text{MA:} \qquad y_t \approx b_1 u_t + b_2 u_{t-1} + \cdots + b_k u_{t-k+1}$$

We then used least-squares to find the coefficients $b_1, \ldots, b_k$. What if we didn't have access to the inputs at all, and we were asked to predict future y values based *only* on the previous y values? One way to do this is by using an *autoregressive* (AR) model, where each output is approximated by a linear combination of the l most recent outputs (excluding the present one):

$$\text{AR:} \qquad y_t \approx a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_\ell y_{t-\ell}$$

Of course, if the inputs contain pertinent information, we shouldn't expect the AR method to outperform the MA method!

In [139]:
```
# Load the data file (ref: Boyd/263)
raw = readcsv("uy_data.csv");
u = raw[:,1]; # inputs
y = raw[:,2]; # true output
T = length(u)

# plot the u and y data
using PyPlot
figure(figsize=(12,4))
plot([u y],".-");
legend(["input u", "true output y"], loc="lower right");
```



a) Using the same dataset from class **uy_data.csv**, plot the true y, and on the same axes, also plot the estimated $\hat{y}$ using the MA model and the estimated $\hat{y}$ using the AR model. Use $k = l = 5$ for both models. To quantify the difference between estimates, also compute $\|y - \hat{y}\|$ for both cases.

In [140]:

```
# Moving-average (MA)
k = 5
A_MA = zeros(T, k)

for i = 1:k
    A_MA[i:end, i] = u[1:end-i+1]
end

wopt_MA = A_MA\y
yest_MA = A_MA*wopt_MA
println(wopt_MA)

# compute the error that the moving-average model makes
MaxWidth = 40
err_MA = zeros(MaxWidth)
for width = 1:MaxWidth
    AMA = zeros(T,width)
    for i = 1:width
        AMA[i:end,i] = u[1:end-i+1]
    end
    wMA = AMA\y
    yMAest = AMA*wMA
    err_MA[width] = norm(y-yMAest)
end
```

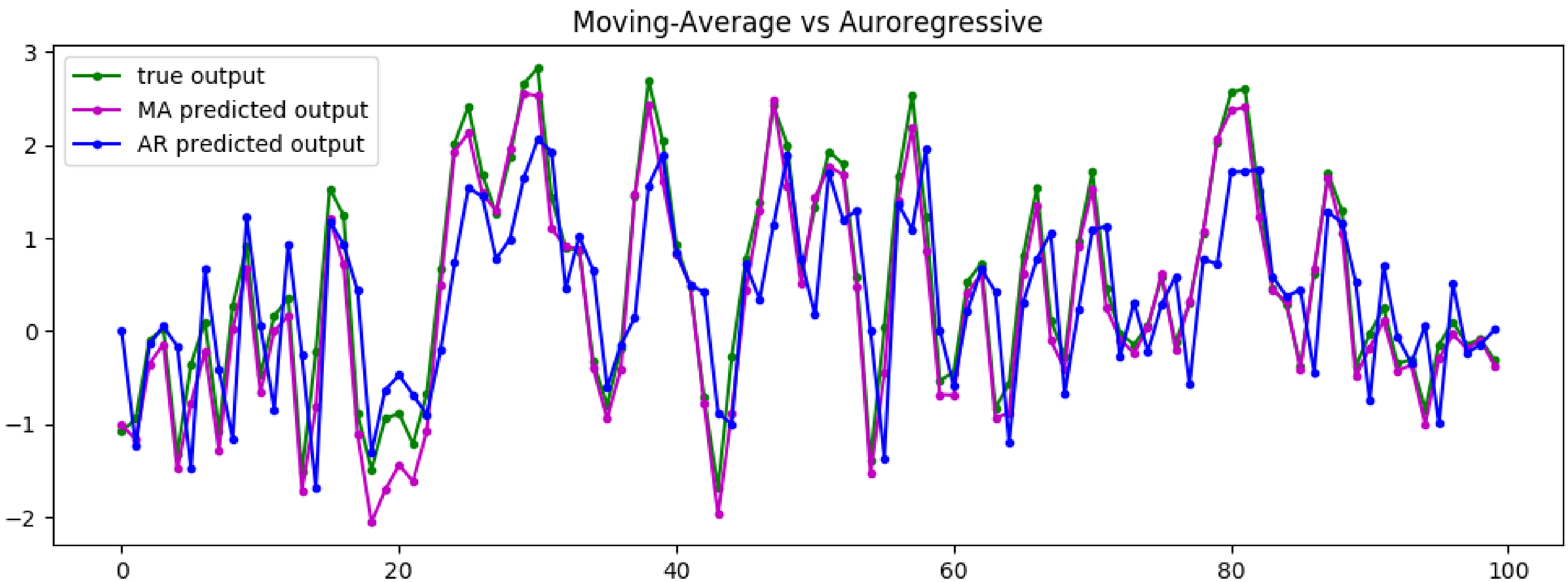
```
[1.1012,0.528947,0.262297,0.0521686,0.00421062]
```

In [141]:

```
# Autoregressive (AR)
l = 5
A_AR = zeros(T, l)

for i = 1:l
    A_AR[i+1:end, i] = y[1:end-i]
end

wopt_AR = A_AR\y
yest_AR = A_AR*wopt_AR
println(wopt_AR)

# compute the error that the autoregressive model makes
MaxWidth = 40
err_AR = zeros(MaxWidth)
for width = 1:MaxWidth
    AAR = zeros(T,width)
    for i = 1:width
        AAR[i+1:end, i] = y[1:end-i]
    end
    wAR = AAR\y
    yARest = AAR*wAR
    err_AR[width] = norm(y-yARest)
end
```

```
[1.1482,-0.876969,0.620235,-0.289534,0.139]
```

In [142]:
```
using PyPlot

figure(figsize=(12,4))
plot(y, "g.-", yest_MA, "m.-", yest_AR, "b.-")
legend(["true output", "MA predicted output", "AR predicted output"], loc="top left");
title("Moving-Average vs Auroregressive")

println("MA estimated error: ", norm(yest_MA - y))
println("AR estimated error: ", norm(yest_AR - y))
```




```
MA estimated error: 2.460854388269911
AR estimated error: 7.436691765656793
```

b) Yet another possible modeling choice is to combine both AR and MA. Unsurprisingly, this is called the autoregressive moving average (ARMA) model:

$$\text{ARMA:} \qquad y_t \approx a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_\ell y_{t-\ell} + b_1 u_t + b_2 u_{t-1} + \cdots + b_k u_{t-k+1}$$

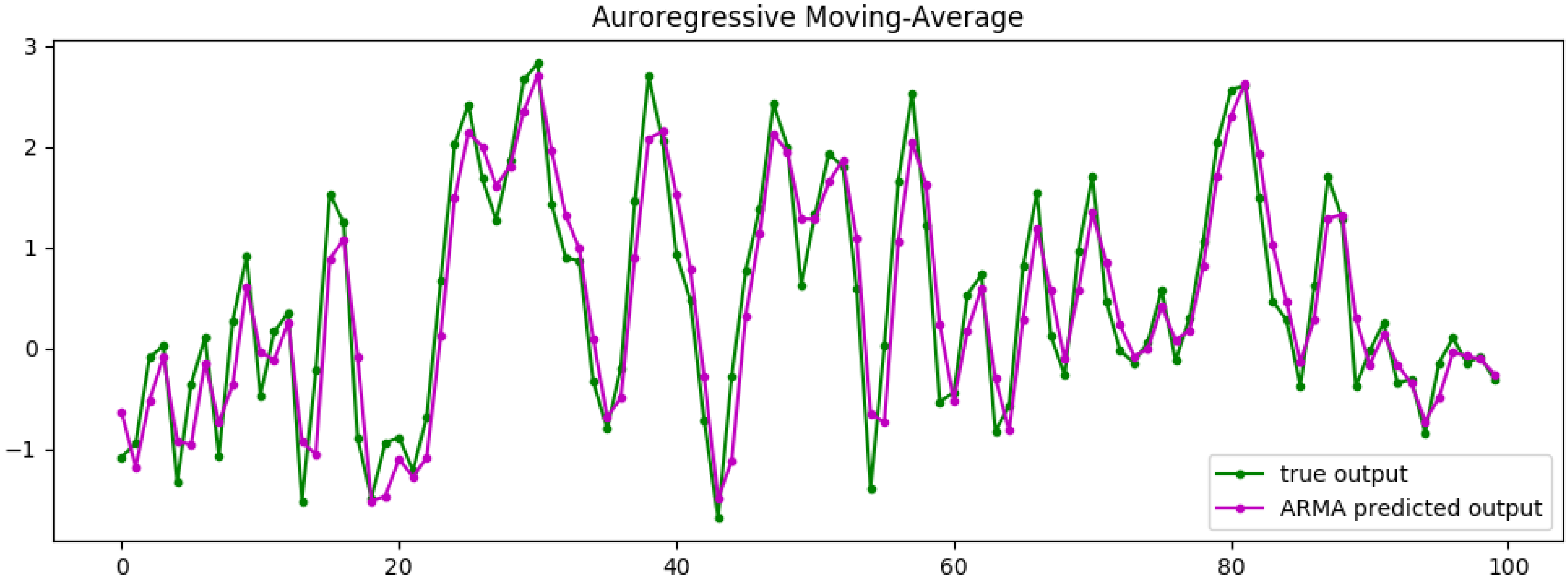
Solve the problem once more, this time using an ARMA model with $k = l = 1$. Plot y and $\hat{y}$ as before, and also compute the error $\|y - \hat{y}\|$.

In [143]:

```
# Autoregressive Moving-average (ARMA)
k = l = 1
A_MA = zeros(T, k)
A_AR = zeros(T, l)

for i = 1:k
    A_MA[i:end, i] = u[1:end-i+1]
end

for i = 1:l
    A_AR[i+1:end, i] = y[1:end-i]
end

A_ARMA = A_AR + A_MA

wopt_ARMA = A_ARMA\y
yest_ARMA = A_ARMA*wopt_ARMA
println(wopt_ARMA)

# compute the error that the autoregressive moving-average model makes
MaxWidth = 40
err_ARMA = zeros(MaxWidth)
for width = 1:MaxWidth
    AMA = zeros(T,width)
    for i = 1:width
        AMA[i:end, i] = u[1:end-i+1]
    end
    AAR = zeros(T,width)
    for i = 1:width
        AAR[i+1:end, i] = y[1:end-i]
    end
    AARMA = AAR + AMA
    wARMA = AARMA\y
    yARMAest = AARMA*wARMA
    err_ARMA[width] = norm(y-yARMAest)
end
```

```
[0.696848]
```

In [144]:
```
using PyPlot

figure(figsize=(12,4))
plot(y,"g.-",yest_ARMA,"m.-")
legend(["true output", "ARMA predicted output"], loc="lower right");
title("Auroregressive Moving-Average");
println("ARMA estimated error: ", norm(yest_ARMA - y))
```




```
ARMA estimated error: 3.9412385247036528
```

In [145]:
```
figure(figsize=(8,3))
title("Error as a function of window size")
plot(1:MaxWidth, err_MA, "m.-")
plot(1:MaxWidth, err_AR, "b.-")
plot(1:MaxWidth, err_ARMA, "y.-")
xlabel("window size")
ylabel("error")
legend(["MA", "AR", "ARMA"],loc="top right",fontsize=10)
;
```