T-61.3040 Statistical Signal Modeling

Autumn 2011

Amaury Lendasse & Yoan Miche

Today's Topics (11.10)

- Signal Modeling (about time...)
- Least Squares Method
- Padé Approximation
- Prony's Method
- Modeling for finite sequences

Signal Modeling

- Why model signals?
- Think about $x(n) = A\sin(n\omega_0 + \phi)$, the Harmonic process
- A bit less data in transfering (A, ω_0, ϕ) and reconstructing at destination than sending all values
- Not obvious which is best: tradeoff between accuracy and efficiency (for general case, where $x(n)$ has no exact expression)

Signal Modeling

- Also for prediction: if you know $x(0), \ldots, x(N-1)$ and want $x(N)$: modeling $x(n)$ with $n \in [0, N-1]$ gives possibility to predict (how to do it accurately is the problem...)
- Two steps in modeling:
  - Choose an "*appropriate*" parametric form for the model
  - Find the model parameters that *best* fit the data records you have

Signal Modeling

- How to do this? Usually, stick to models with well-known properties, such as ARMA type, sum of weighted sinusoids...
- Otherwise, finding model parameters might be intractable (remember you have to optimize a cost function to find them)
- Cost/Error of the modelisation: $e'(n) = x(n) - \hat{x}(n)$
- Remember another type of cost?

Least Squares Method

Also called "Direct Method"

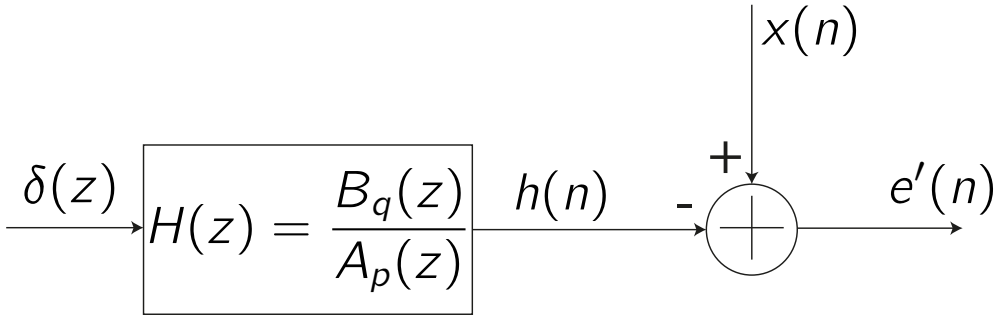

Figure: Block Diagram of the Direct Method

Least Squares Method

- Consider modeling of a deterministic signal $x(n)$ as the unit sample response of a LSI filter with rational system function of the form

$$H(z) = \frac{B_q(z)}{A_p(z)} = \frac{\sum_{k=0}^{q} b_q(k) z^{-k}}{1 + \sum_{k=1}^{p} a_p(k) z^{-k}}$$

- Assume also that $x(n) = 0$ for $n < 0$ (on loss of generality, just shift the signal) and $h(n)$ causal

Least Squares Method

- Criterion to be minimized for this method is based on squares of error:

$$\varepsilon_{LS} = \sum_{n=0}^{\infty} \left| e'(n) \right|^2$$

- Minimizing the LS cost with such a system function means

$$\begin{cases} \frac{\partial \epsilon_{LS}}{\partial a_p^*(k)} = 0, & \forall k \in [\![1, p]\!] \\ \frac{\partial \epsilon_{LS}}{\partial b_q^*(k)} = 0 & \forall k \in [\![0, q]\!] \end{cases}$$

Least Squares Method

- Writing the expression of the LS error in the frequency domain, we get the set of equations ($k = 1, \dots p$ in the first, $k = 0, \dots, q$ in the second)

$$\begin{cases} \frac{\partial \epsilon_{LS}}{\partial a_p^*(k)} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[X(e^{j\omega}) - \frac{B_p(e^{j\omega})}{A_q(e^{j\omega})} \right] \frac{B_q^*(e^{j\omega})}{\left[A_p^*(e^{j\omega})\right]^2} e^{jk\omega} d\omega = 0 \\ \frac{\partial \epsilon_{LS}}{\partial b_q^*(k)} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[X(e^{j\omega}) - \frac{B_p(e^{j\omega})}{A_q(e^{j\omega})} \right] \frac{e^{jk\omega}}{A_p^*(e^{j\omega})} d\omega = 0 \end{cases}$$

- And we have ourselves $p + q + 1$ nonlinear equations

Least Squares Method

- Usually intractable, or needs iterative methods: Newton's method, steepest descent...
- We will use indirect methods instead of trying to solve the direct one

Padé Approximation

- In this case we will end up with a set of linear equations
- With same assumptions on $x(n)$ and the system function (p poles and q zeros, same form as before, $x(n)$ modeled as a unit response to a LSI causal filter)
- We have $p + q + 1$ degrees of freedom on the filter design
- So, we should be able to make the filter fit exactly the signal for $p + q + 1$ values out of n, right?

Padé Approximation: A simple example

An example:

- With a causal first order all pole filter such as

$$H(z) = \frac{b(0)}{1 + a(1)z^{-1}}$$

- The unit sample response of the filter is then

$$h(n) = b(0)\left[-a(1)\right]^n u(n)$$

Padé Approximation: A simple example

- So if we want $h(n) = x(n)$ for $n = 0, 1$, we need

$$\begin{cases} x(0) &= b(0) \\ x(1) &= -b(0)a(1) \end{cases}$$

- Meaning, if $x(0) \neq 0$, that the model is

$$H(z) = \frac{x^2(0)}{x(0) - x(1)z^{-1}}$$

Padé Approximation: A simple example

Another example, higher degree:

- Have

$$H(z) = \frac{b(0) + b(1)z^{-1}}{1 + a(1)z^{-1}}$$

- Leads to

$$\begin{cases} x(0) &= b(0) \\ x(1) &= -b(0)a(1) + b(1) \\ x(2) &= b(0)a^2(1) - b(1)a(1) \end{cases}$$

Padé Approximation: A simple example

- And finally, solving by substitution

$$\begin{cases} a(1) & = -\frac{x(2)}{x(1)} \\ b(0) & = x(0) \\ b(1) & = x(1) - x(0)\frac{x(2)}{x(1)} \end{cases}$$

  assuming $x(1) \neq 0$

- Did I lie? These last equations are not exactly linear...

Padé Approximation

- Padé Approximation aims at getting the right coefficients while reformulating the problem to only have linear equations to solve
- Start by expressing the system function

$$H(z) = \frac{B_q(z)}{A_p(z)} = \frac{\sum_{k=0}^{q} b_q(k) z^{-k}}{1 + \sum_{k=1}^{p} a_p(k) z^{-k}}$$

as

$$H(z) A_p(z) = B_q(z)$$

Padé Approximation

- And express it in the time domain as the convolution

$$h(n) + \sum_{k=1}^{p} a_p(k)h(n-k) = b_q(n)$$

with $h(n) = 0$ for $n < 0$ and $b_q(n) = 0$ for $n < 0$ and $n > q$

- Now we want $x(n) = h(n)$ over $[\![0, p+q]\!]$, so we write the equations

$$x(n) + \sum_{k=1}^{p} a_p(k)x(n-k) = \begin{cases} b_q(n) & , n \in [\![0, q]\!] \\ 0 & , n \in [\![q+1, q+p]\!] \end{cases}$$

Padé Approximation

Which will be handier in matrix form (why? later...)

$$\left[\begin{array}{cccc} x(0) & 0 & \cdots & 0 \\ x(1) & x(0) & \cdots & 0 \\ x(2) & x(1) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ \hline x(q+1) & x(q) & \cdots & x(q-p+1) \\ \vdots & \vdots & \cdots & \vdots \\ x(q+p) & x(q+p-1) & \cdots & x(q) \end{array}\right] \left[\begin{array}{c} 1 \\ a_p(1) \\ a_p(2) \\ \vdots \\ a_p(p) \end{array}\right] = \left[\begin{array}{c} b_q(0) \\ b_q(1) \\ b_q(2) \\ \vdots \\ b_q(q) \\ \hline 0 \\ \vdots \\ 0 \end{array}\right]$$

Padé Approximation

To solve this, we first solve the "lower part" of this system to get the $a_p(k)$:

$$\begin{bmatrix} x(q+1) & x(q) & \cdots & x(q-p+1) \\ x(q+2) & x(q+1) & \cdots & x(q-p+2) \\ \vdots & \vdots & \cdots & \vdots \\ x(q+p) & x(q+p-1) & \cdots & x(q) \end{bmatrix} \begin{bmatrix} 1 \\ a_p(1) \\ \vdots \\ a_p(p) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Padé Approximation

Which can be expanded and rewritten as

$$\begin{bmatrix} x(q) & x(q-1) & \cdots & x(q-p+1) \\ x(q+1) & x(q) & \cdots & x(q-p+2) \\ \vdots & \vdots & \cdots & \vdots \\ x(q+p-1) & x(q+p-2) & \cdots & x(q) \end{bmatrix} \begin{bmatrix} a_p(1) \\ a_p(2) \\ \vdots \\ a_p(p) \end{bmatrix} = - \begin{bmatrix} x(q+1) \\ x(q+2) \\ \vdots \\ x(q+p) \end{bmatrix}$$

which we express as $\mathbf{X}_q \mathbf{a}_p = -\mathbf{x}_{q+1}$

Padé Approximation

And $\mathbf{X}_q$ is a nonsymmetric Toeplitz matrix, which means that depending on how $\mathbf{X}_q$ is structured, we have solutions or not:

- $\mathbf{X}_q$ not singular: Then $\mathbf{X}_q^{-1}$ exists and the solution is uniquely determined by $\mathbf{a}_p = -\mathbf{X}_q^{-1}\mathbf{x}_{q+1}$
- $\mathbf{X}_q$ singular and a solution exists: The solution is not unique. Might want to choose the one with smallest amount of non-zero values
- $\mathbf{X}_q$ singular and no solutions: We assumed in the model formulation, that $a_p(0) = 1$ (or was non-zero and could be normalized to be 1). Must be false, so we have in fact $a_p(0) = 0$, and then $\mathbf{X}_q\mathbf{a}_p = 0$ which has a nonzero solution

Padé Approximation

- And now that we have the $\mathbf{a}_p$, let us get the $\mathbf{b}_q$
- Taking the "upper part" of the system, we have

$$\begin{bmatrix} x(0) & 0 & \cdots & 0 \\ x(1) & x(0) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ x(q) & x(q-1) & \cdots & x(q-p) \end{bmatrix} \begin{bmatrix} 1 \\ a_p(1) \\ \vdots \\ a_p(p) \end{bmatrix} = \begin{bmatrix} b_q(0) \\ b_q(1) \\ \vdots \\ b_q(q) \end{bmatrix}$$

  which in matrix form is this time $\mathbf{X}_0\mathbf{a}_p = \mathbf{b}_q$
- And you just have to do a matrix multiplication to get the $\mathbf{b}_q$

Padé Approximation: Example

Now an example:

- Take a signal of which you have values

$$\mathbf{x} = [1, 1.5, 0.75, 0.375, 0.1875, 0.0938]^T$$

- Say we'd like two filters:
  1. A second order all pole filter
  2. A model with just one pole and one zero

Padé Approximation: Example

Let's start with the first one: Second order all pole

- We have to solve:

$$\begin{bmatrix} x(0) & 0 & 0 \\ x(1) & x(0) & 0 \\ x(2) & x(1) & x(0) \end{bmatrix} \begin{bmatrix} 1 \\ a(1) \\ a(2) \end{bmatrix} = \begin{bmatrix} b(0) \\ 0 \\ 0 \end{bmatrix}$$

- Take the last two equations and substitute:

$$\begin{bmatrix} 1 & 0 \\ 1.5 & 1 \end{bmatrix} \begin{bmatrix} a(1) \\ a(2) \end{bmatrix} = - \begin{bmatrix} 1.5 \\ 0.75 \end{bmatrix}$$

which gives $a(1) = -1.5$ and $a(2) = 1.5$

Padé Approximation: Example

And I think finding the $b(0)$ should not be difficult

- So we get

$$H(z) = \frac{1}{1 - 1.5z^{-1} + 1.5z^{-2}}$$

- And the approximation given by the model is

$$\hat{\mathbf{x}} = [1, 1.5, 0.75, -1.125, -2.8125, -2.5312]^T$$

- And we wanted

$$\mathbf{x} = [1, 1.5, 0.75, 0.375, 0.1875, 0.0938]^T$$

but only enforced on the first three values, due to filter design

Padé Approximation: Example

Second one: One pole, one zero

- We have

$$H(z) = \frac{b(0) + b(1)z^{-1}}{1 + a(1)z^{-1}}$$

- So the Padé equations are

$$\begin{bmatrix} x(0) & 0 \\ x(1) & x(0) \\ x(2) & x(1) \end{bmatrix} \begin{bmatrix} 1 \\ a(1) \end{bmatrix} = \begin{bmatrix} b(0) \\ b(1) \\ 0 \end{bmatrix}$$

- So $a(1) = -\frac{x(2)}{x(1)} = -0.5$

Padé Approximation: Example

And for the **b** coefficients, we solve

$$\begin{bmatrix} b(0) \\ b(1) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1.5 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -0.5 \end{bmatrix} = \begin{matrix} 1 \\ 1 \end{matrix}$$

- And the model is then

$$H(z) = \frac{1 + z^{-1}}{1 - 0.5z^{-1}}$$

- Which matches the required values (I'll let you check that by writing the inverse transform of the model)
- It was meant to happen here, for **x** was designed as the unit sample response of a LSI filter

Padé Approximation: Conclusion

In conclusion on Padé's approximation:

- A model created using the Padé's approximation will always produce an exact fit over the data in the interval $[\![0, p+q]\!]$ if $\mathbf{X}_q$ is not singular
- Since we do not consider the data outside of that interval, likely we don't have an accurate model outside that interval
- Padé's approximation forces the model to fit over a limited range of values, with no other constraints: model might be unstable (no guarantees)

Prony's Method

- Relax the constraint to match exactly the values on the interval $[\![0, p+q]\!]$, in order to better behave over the whole range
- Will also have a set of linear equations to solve to have filter coefficients
- We still want to minimize the error

$$e'(n) = x(n) - h(n)$$

where $h(n)$ is the unit sample response of the filter that is supposed to match $x(n)$

- Which can also be written $E'(z) = X(z) - \frac{B_q(z)}{A_p(z)}$

Prony's Method

- Now if we use the idea from the Padé approximation, and multiply this expression by $A_p(z)$, we have another error, linear with the filter coefficients

$$E(z) = A_p(z)E'(z) = A_p(z)X(z) - B_q(z)$$

- Going back to the time domain, we have

$$e(n) = a_p(n) * x(n) - b_q(n) = \hat{b}_q(n) - b_q(n)$$

Prony's Method

And since $b_q(n) = 0, \forall n > q$, we can write the error for all values of n as

$$e(n) = \begin{cases} x(n) + \sum_{l=1}^{p} a_p(l)x(n-l) - b_q(n) & , n = 0, \ldots, q \\ x(n) + \sum_{l=1}^{p} a_p(l)x(n-l) & , n > q \end{cases}$$

Prony's Method

- Now, with the Padé approximation idea, we would set $e(n) = 0$ for $n = 0, \ldots, p + q$
- With Prony's, we first find the **a** coefficients that minimize the least squares error

$$\varepsilon_{p,q} = \sum_{n=q+1}^{\infty} |e(n)|^2 = \sum_{n=q+1}^{\infty} \left| x(n) + \sum_{l=1}^{p} a_p(l) x(n-l) \right|^2$$

Prony's Method

Using partial derivatives w.r.t. $a_p^*(k)$, we get the minimum of $\varepsilon_{p,q}$ as the solution of

$$\sum_{n=q+1}^{\infty} e(n)x^*(n-k) = 0, k = 1, \ldots, p$$

Prony's Method

Substituting the previous expression of $e(n)$ for all the n in this equation, we have finally

$$\sum_{l=1}^{p} a_p(l) r_x(k,l) = -r_x(k,0), k = 1, \ldots, p$$

with $r_x(k,l) = \sum_{n=q+1}^{\infty} x(n-l)x^*(n-k)$ which is similar to an autocorrelation sequence

Prony's Method

Now, the previous equation is a set of equations, that can be written in matrix form as

$$\begin{bmatrix} r_x(1,1) & r_x(1,2) & \cdots & r_x(1,p) \\ r_x(2,1) & r_x(2,2) & \cdots & r_x(2,p) \\ \vdots & \vdots & \cdots & \vdots \\ r_x(p,1) & r_x(p,2) & \cdots & r_x(p,p) \end{bmatrix} \begin{bmatrix} a_p(1) \\ a_p(2) \\ \vdots \\ a_p(p) \end{bmatrix} = - \begin{bmatrix} r_x(1,0) \\ r_x(2,0) \\ \vdots \\ r_x(p,0) \end{bmatrix}$$

or $\mathbf{R}_x \mathbf{a}_p = -\mathbf{r}_x$

Prony's Method

- Now, with an infinite column length matrix $\mathbf{X}_q$ defined as

$$\mathbf{X}_q = \begin{bmatrix} x(q) & x(q-1) & \cdots & x(q-p+1) \\ x(q+1) & x(q) & \cdots & x(q-p+2) \\ \vdots & \vdots & \cdots & \vdots \end{bmatrix}$$

- We can express $\mathbf{R}_x = \mathbf{X}_q^H \mathbf{X}_q$ and $\mathbf{r}_x = \mathbf{X}_q^H \mathbf{x}_{q+1}$ with $\mathbf{x}_{q+1} = [x(q+1), x(q+2), \ldots]^T$ (also infinite length)
- And then the Prony equations become

$$\left(\mathbf{X}_q^H \mathbf{X}_q\right) \mathbf{a}_p = -\mathbf{X}_q^H \mathbf{x}_{q+1}$$

Prony's Method

- Meaning that if $\mathbf{R}_x$ is nonsingular, we have the coefficients

$$\mathbf{a}_p = -\left(\mathbf{X}_q^H \mathbf{X}_q\right)^{-1} \mathbf{X}_q^H \mathbf{x}_{q+1}$$

- Injecting these coefficients in the error, we have the minimum $\epsilon_{p,q}$ of the original error $\varepsilon_{p,q}$ as

$$\epsilon_{p,q} = r_x(0,0) + \sum_{k=1}^{p} a_p(k) r_x(0,k)$$

Prony's Method

- Now, to find the **b** coefficients, we end up with the same as for Padé's approximation: setting the error to zero and solving

$$e(n) = a_p(n) * x(n) - b_q(n) = 0$$

- So, the **b** coefficients are obtained by the set of equations

$$b_q(n) = x(n) + \sum_{k=1}^{p} a_p(k)x(n-k)$$

- Which is straightforward to obtain

Prony's Method

An example:

- Let's have a signal $x(n)$ consisting of a pulse of length N, as

$$x(n) = \begin{cases} 1 & , n = 0, \ldots, N-1 \\ 0 & \text{otherwise} \end{cases}$$

- And we'll model it as the unit sample response of a LSI filter with one pole and one zero:

$$H(z) = \frac{b(0) + b(1)z^{-1}}{1 + a(1)z^{-1}}$$

Prony's Method

- So, with $p = 1$, the Prony equations in a_p are

$$a(1)r_x(1,1) = -r_x(1,0)$$

- And we can calculate $r_x(1,1)$ and $r_x(1,0)$ easily:

$$\begin{aligned} r_x(1,1) &= \sum_{n=2}^{\infty} x^2(n-1) = N-1 \\ r_x(1,0) &= \sum_{n=2}^{\infty} x(n)x(n-1) = N-2 \end{aligned}$$

Prony's Method

- So the denominator of $H(z)$, $A(z)$, becomes

$$A(z) = 1 - \frac{N-2}{N-1} z^{-1}$$

- And for the b_q coefficients, we have the equations

$$\begin{aligned} b(0) &= x(0) = 1 \\ b(1) &= x(1) + a(1)x(0) = 1 - \tfrac{N-2}{N-1} = \tfrac{1}{N-1} \end{aligned}$$

Prony's Method

- So finally

$$H(z) = \frac{1 + \frac{1}{N-1} z^{-1}}{1 - \frac{N-2}{N-1} z^{-1}}$$

- And the minimum squared error can be computed as

$$\epsilon_{1,1} = r_x(0,0) + a(1) r_x(0,1)$$

- Since $r_x(0,0) = N - 2$, we have $\epsilon_{1,1} = \frac{N-2}{N-1}$

Further Reading on this subject

- Shank's method: Extend the LS constraint of Padé over the entire series of records
- All pole using Shank's method
- Please have a read on the course webpage "Additional Reading" section (not mandatory): Link to PDF excerpt on Shank's Method

Finite Data records

- Prony's method assumes $x(n)$ known for all n
- What if we just have a window of values on $x(n)$?
  - Autocorrelation Method: "window" our process $x(n)$ by a rectangular window which sets $x(n)$ to zero outside of the records we have
  - Covariance Method: minimize an error that does not depend on the values of $x(n)$ outside of the records we have

Autocorrelation Method

Suppose we have a signal $x(n)$ (eventually complex) known only on the interval $[0, N]$ and that we want to approximate $x(n)$ by an all-pole model

$$H(z) = \frac{b(0)}{1 + \sum_{k=1}^{p} a_p(k) z^{-k}}$$

Autocorrelation Method

With Prony's method, we minimize

$$\varepsilon_p = \sum_{n=0}^{\infty} |e(n)|^2$$

with

$$e(n) = x(n) + \sum_{k=1}^{p} a_p(k) x(n-k)$$

Autocorrelation Method

- But if we don't know $x(n)$ outside of $[0, N]$, we can't evaluate $e(n)$ for $n < p$ or $n > N$
- So we create new signal, $\tilde{x}(n)$ based on $x(n)$, by applying a rectangular window:

$$\tilde{x}(n) = x(n)w(n)$$

with

$$w(n) = \begin{cases} 1 & , n = 0, \ldots, N \\ 0 & , \text{otherwise} \end{cases}$$

Autocorrelation Method

- Then we can use Prony's method to find an all-pole model for $\tilde{x}(n)$
- The Prony equations are identical, except that the autocorrelation is that of $\tilde{x}(n)$ instead of $x(n)$, i.e. replace the usual $r_x(k)$ by $r_{\tilde{x}}(k)$:

$$r_{\tilde{x}}(k) = \sum_{n=0}^{\infty} \tilde{x}(n)\tilde{x}^*(n-k) = \sum_{n=k}^{N} x(n)x^*(n-k), k = 0, \ldots, p$$

Covariance Method

- Covariance Method
- What if you don't want to force the signal to be zero outside for solving?
- Covariance method usually gives more accurate models than autocovariance one, but lose the Toeplitz structure of the equations (and hence the fast ways to solve)

Covariance Method

- Again with an all-pole model
- Different error ε_p^C to minimize, that only uses the known values of $e(n)$ with no assumptions on the other ones (if $x(n)$ is known for $[0, N]$, then $e(n)$ can be known for $[p, N]$)

$$\varepsilon_p^C = \sum_{n=p}^{N} |e(n)|^2$$

- Finding the all-pole coefficients minimizing this error is known as the Covariance Method

Covariance Method

It leads to the same set of normal equations as for Prony's method, called here the covariance normal equations

$$\begin{bmatrix} r_x(1,1) & r_x(1,2) & \cdots & r_x(1,p) \\ r_x(2,1) & r_x(2,2) & \cdots & r_x(2,p) \\ \vdots & \vdots & \vdots & \vdots \\ r_x(p,1) & r_x(p,2) & \cdots & r_x(p,p) \end{bmatrix} \begin{bmatrix} a_p(1) \\ a_p(2) \\ \vdots \\ a_p(3) \end{bmatrix} = - \begin{bmatrix} r_x(1,0) \\ r_x(2,0) \\ \vdots \\ r_x(p,0) \end{bmatrix}$$

with the autocorrelation sequence $r_x(k,l)$ expressed as

$$r_x(k,l) = \sum_{n=p}^{N} x(n-l)x^*(n-k)$$

Covariance Method

- And the minimum covariance modeling error ϵ_p^C is

$$\epsilon_p^C = r_x(0,0) + \sum_{k=1}^{p} a_p(k) r_x(0,k)$$

- Covariance normal equations are not Toeplitz, so no easy/fast solving in the general case (fast solving algorithms exist, but complicated)

Summary

- Least Squares Method
- Padé Approximation
- Prony's Method
- Autocorrelation Method
- Covariance Method

Next time

- More examples for AR, MA, ARMA models
- Power Spectrum estimation
- A look at the Levinson-Durbin recursion to solve Toeplitz equations
- A look at lattice filters