ELEC system identification workshop

Exercises

Ivan Markovsky

Outline

Introduction

Behavioral approach

Subspace methods

Optimization methods

SLRA package

Line fitting

problem: fit points $d_1, \dots, d_N \in \mathbb{R}^2$ by a line

1. find condition for existence of a line (any line in $\mathbb{R}^2$) that passes through the points

2. how would you test the condition in MATLAB?

3. implement a method for exact line fitting

Solution for part 1

the points $d_i = (a_i, b_i)$, $i = 1, \dots, N$ lie on line

$$\Updownarrow$$

there is $(R_1, R_2, R_3) \neq 0$, such that
$R_1 a_i + R_2 b_i + R_3 = 0$, for $i = 1, \dots, N$

$$\Updownarrow$$

there is $(R_1, R_2, R_3) \neq 0$, such that

$$\begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix} \begin{bmatrix} a_1 & \cdots & a_N \\ b_1 & \cdots & b_N \\ 1 & \cdots & 1 \end{bmatrix} = 0$$

$$\Updownarrow$$

$$\text{rank}\left(\begin{bmatrix} a_1 & \cdots & a_N \\ b_1 & \cdots & b_N \\ 1 & \cdots & 1 \end{bmatrix} \right) \leq 2$$

Solution for part 2

given matrix `d` which columns are the data points

exact fitting condition:

```
N    = size(d, 2)
dext = [d; ones(1, N)];

if (rank(dext) < 3)
   disp('exact fit exists')
else
   disp('exact fit does not exist')
end
```

Solution for part 3

given matrix `d` which columns are the data points

exact fitting method:

```
N    = size(d, 2)
dext = [d; ones(1, N)];

r = null(dext')';
```

Note

$\mathscr{B} = \{\, d \mid Rd = 0 \,\}$ — linear static model

$\mathscr{B} = \{\, d \mid R \left[\begin{smallmatrix} d \\ 1 \end{smallmatrix} \right] = 0 \,\}$ — affine static model

in exact modeling

$$\begin{gathered}\text{affine fitting} \\ \Updownarrow \\ \text{data centering + linear modeling}\end{gathered}$$

homework: is the same true in approximate modeling?

Conic section fitting

problem: fit points $d_1, \ldots, d_N \in \mathbb{R}^2$ by conic section

$$\mathscr{B}(S,u,v) = \{\, d \in \mathbb{R}^2 \mid d^\top S d + u^\top d + v = 0 \,\}$$

1. find condition for existence of an exact fit

2. propose numerical method for exact fitting

3. implement the method and test it on the data

$$d_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad d_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad d_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad d_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

Solution for part 1

the points $d_i = (a_i, b_i)$, $i = 1, \ldots, N$ lie on conic section

$$\Updownarrow$$

$\exists\ S = S^\top$, u, v, at least one of them nonzero, such that
$d_i^\top S d_i + u^\top d_i + v = 0$, for $i = 1, \ldots, N$

$$\Updownarrow$$

there is $(s_{11}, s_{12}, s_{22}, u_1, u_2, v) \neq 0$, such that

$$\begin{bmatrix} s_{11} & 2s_{12} & u_1 & s_{22} & u_2 & v \end{bmatrix} \begin{bmatrix} a_1^2 & \cdots & a_N^2 \\ a_1 b_1 & \cdots & a_N b_N \\ a_1 & \cdots & a_N \\ b_1^2 & \cdots & b_N^2 \\ b_1 & \cdots & b_N \\ 1 & \cdots & 1 \end{bmatrix} = 0$$

Solution for part 1 (continued)

the points $d_i = (a_i, b_i)$, $i = 1, \dots, N$ lie on conic section

$$\Updownarrow$$

$$\text{rank}\left(\begin{bmatrix} a_1^2 & \cdots & a_N^2 \\ a_1 b_1 & \cdots & a_N b_N \\ a_1 & \cdots & a_N \\ b_1^2 & \cdots & b_N^2 \\ b_1 & \cdots & b_N \\ 1 & \cdots & 1 \end{bmatrix}\right) \leq 5$$

```
f = @(a, b) [a .^ 2; a .* b; a; b .^ 2; b;
             ones(size(a))];
```

Solution for part 2 and 3

finding exact models

```
R = null(f(d(1, :), d(2, :))')';
```

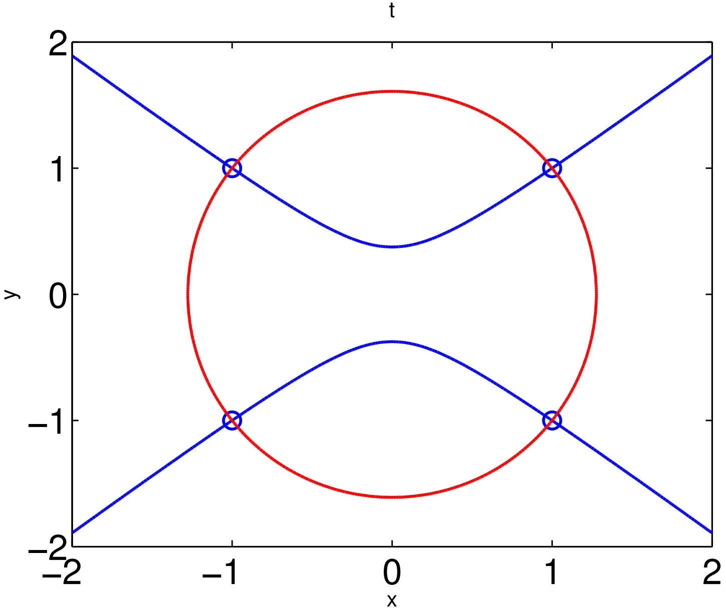
plotting model

```
function H = plot_model(th, f, ax, c)
H = ezplot(@(a, b) th * f(a, b), ax);
for h = H', set(h, 'color', c, 'linewidth', 2);
```

show results

```
plot(d(1, :), d(2, :), 'o', 'markersize', 12)
ax = 2 * axis;
for i = 1:size(R, 1)
  hold on, plot_model(R(i, :), f, ax, c(i));
end
```

Solution for part 2 and 3 (continued)



Recursive sequence fitting

problem: fit $w = \big(w(1), \dots, w(T)\big)$ by model

$$\mathscr{B} = \{\, w \mid R_0 w + R_1 \sigma w + \cdots + R_\ell \sigma^\ell w = 0 \,\}$$

1. find condition for existence of an exact fit
   first, with, and then, without knowledge of ℓ

2. propose numerical method for exact fitting
   find the smallest ℓ, for which exact model exists

3. implement the method and test it on the data

$$(1, 2, 4, 7, 13, 24, 44, 81)$$

Solution for part 1

$$w = \big(w(1), \dots, w(T)\big) \in \mathscr{B}$$

$$\mathscr{B} = \{\, w \mid R_0 w + R_1 \sigma w + \cdots + R_\ell \sigma^\ell w = 0 \,\}$$

$$\Updownarrow$$

$$R_0 w(t) + R_1 w(t+1) + \cdots + R_\ell w(t+\ell) = 0$$

$$\text{for } t = 1, \dots, T-\ell$$

$$\Updownarrow$$

$$\operatorname{rank}\big(\mathscr{H}_{\ell+1}(w)\big) \leq \ell$$

$$\mathscr{H}_{\ell+1}(w) := \begin{bmatrix} w(1) & w(2) & \cdots & w(T-\ell) \\ w(2) & w(3) & \cdots & w(T-\ell+1) \\ \vdots & \vdots & & \vdots \\ w(\ell+1) & w(\ell+2) & \cdots & w(T) \end{bmatrix}$$

relation at time $t = 1$

$$R_0 w(1) + R_1 w(2) + \cdots + R_\ell w(\ell + 1) = 0$$

in matrix form:

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \begin{bmatrix} w(1) \\ w(2) \\ \vdots \\ w(\ell+1) \end{bmatrix} = 0$$

relation at time $t = 2$

$$R_0 w(2) + R_1 w(3) + \cdots + R_\ell w(\ell + 2) = 0$$

in matrix form:

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \begin{bmatrix} w(2) \\ w(3) \\ \vdots \\ w(\ell+2) \end{bmatrix} = 0$$

relation at time $t = T - \ell$

$$R_0 w(T-\ell) + R_1 w(T-\ell+1) + \cdots + R_\ell w(T) = 0$$

in matrix form:

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \begin{bmatrix} w(T-\ell) \\ w(T-\ell+1) \\ w(T-\ell+2) \\ \vdots \\ w(T) \end{bmatrix} = 0$$

Solution for part 2 and 3

with ℓ unknown, do the test for $\ell = 1, 2, \dots$

algorithm

```
for ell = 1:ell_max
   if (rank(H(w, ell + 1)) == ell)
      break
   end
end
```

in the example, $\ell = 3$ and $R = \begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix}$

Outline

Checking whether a sequence is trajectory

1. given sequence w and polynomial R, propose method for checking numerically whether $w \in \mathscr{B} = \ker\big(R(\sigma)\big)$

2. implement it in a function `w_in_ker(w, r)`

3. test it on the trajectory

$$w = (u_d, y_d) = \big((0,1),(0,1),(0,1),(0,1)\big)$$

   and the system

$$\mathscr{B} = \ker\big(R(\sigma)\big), \qquad R(z) = \begin{bmatrix} 1 & -1 \end{bmatrix} + \begin{bmatrix} -1 & 1 \end{bmatrix} z$$

Solution for part 1

w$\in \ker\big(R(\sigma)\big)$

$$\iff R(\sigma)w = 0$$

$$\iff R_0 w(t) + R_1 w(t+1) + \cdots + R_\ell w(t+\ell) = 0 \qquad \text{for } t = 1,\dots,T-\ell$$

$$\iff \underbrace{\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell & & & \\ & R_0 & R_1 & \cdots & R_\ell & & \\ & & \ddots & \ddots & & \ddots & \\ & & & R_0 & R_1 & \cdots & R_\ell \end{bmatrix}}_{\mathscr{M}_T(R)\in\mathbb{R}^{p(T-\ell)\times qT}} \underbrace{\begin{bmatrix} w(1) \\ w(2) \\ \vdots \\ w(T) \end{bmatrix}}_{\text{vec}(w)} = 0$$

numerical test: $\|\mathscr{M}_T(R)\text{vec}(w)\| < \varepsilon$ (with tolerance ε)

Another solution for part 1

w$\in \ker\big(R(\sigma)\big)$

$$\begin{aligned} &\iff \quad \mathscr{M}_T(R)\,\mathrm{vec}(w) = 0 \\ &\iff \quad R\mathscr{H}_{\ell+1}(w) = 0 \end{aligned}$$

where

$$\underbrace{\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix}}_{R} \underbrace{\begin{bmatrix} w(1) & w(2) & \cdots & w(T-\ell) \\ w(2) & w(3) & \cdots & \\ \vdots & \vdots & & \vdots \\ w(\ell+1) & w(\ell+2) & \cdots & w(T) \end{bmatrix}}_{\mathscr{H}_{\ell+1}(w)\in\mathbb{R}^{q(\ell+1)\times(T-\ell)}} = 0$$

numerical test: $\|R\mathscr{H}_{\ell+1}(w)\| < \varepsilon$

Solution for part 2

```
function a = w_in_ker(w, r, ell)
a = norm(r * blkhank(w, ell + 1)) < 1e-8;
```

block-Hankel matrix $\mathscr{H}_L(w)$ constructor

```
function H = blkhank(w, i, j)
[q, T] = size(w);
if T < q, w = w'; [q, T] = size(w); end
if nargin < 3, j = T - i + 1; end
H = zeros(i * q, j);
for ii = 1:i
  H(((ii - 1) * q + 1):(ii * q), :) ...
                  = w(:, ii:(ii + j - 1));
end
```

Solution for part 2 (continued)

```
w = [0 0 0 0; 1 1 1 1];
r = [1 -1 -1 1]; ell = 1;
w_in_ker(w, r, 1)
```

homework

use image representation to check

$$w \overset{?}{\in} \operatorname{image}\big(P(\sigma)\big) \qquad (\texttt{w\_in\_im})$$

use state space representation to check

$$w \overset{?}{\in} \mathscr{B}(A, B, C, D) \qquad (\texttt{w\_in\_ss})$$

Transfer function $\mapsto$ kernel representation

1. what model $\mathscr{B}_{\text{tf}}(H)$ is specified by transfer function

$$H(z) = \frac{q(z)}{p(z)} = \frac{q_0 + q_1 z^1 + \cdots + q_\ell z^\ell}{p_0 + p_1 z^1 + \cdots + p_\ell z^\ell}$$

2. find R, such that

$$\mathscr{B}_{\text{tf}}(H) = \ker(R)$$

3. write function `tf2r` converting H (`tf` object) to R
   and function `r2tf` converting R to H

Solution for part 1 and 2

the transfer function H represents model

$$\mathscr{B}_{\mathrm{tf}}(H) = \{\, w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \mid p(\sigma)y = q(\sigma)u \,\}$$

the corresponding kernel representation is

$$\underbrace{\begin{bmatrix} q(\sigma) & -p(\sigma) \end{bmatrix}}_{R(\sigma)} \begin{bmatrix} u \\ y \end{bmatrix} = 0$$

note: $y = \mathscr{Z}^{-1}\big(H\mathscr{Z}(u)\big)$ assumes zero initial conditions

homework: include initial conditions in $y = \mathscr{Z}^{-1}\big(H\mathscr{Z}(u)\big)$

Solution for part 3

```
function r = tf2r(H)
[Q P] = tfdata(tf(H), 'v');
R = vec(fliplr([Q; -P]))';

function H = r2tf(R)
Q =   fliplr(R(1:2:end));
P = - fliplr(R(2:2:end));
H = tf(Q, P, -1);
```

note: MATLAB uses descending order of coefficients

Initial conditions specification by trajectory

```
LSIM(SYS,U,T,X0) specifies the initial
state vector X0 at time T(1)
  (for state-space models only).
```

problem: given minimal $\mathscr{B} = \mathscr{B}(A, B, C, D) \in \mathscr{L}_{\mathrm{m},\ell}$

1. show that $\underbrace{\big(w(-\ell+1), \dots, w(0)\big)}_{w_\mathrm{p}} \in \mathscr{B}$ determines $x(0)$

2. explain how to use w_p to "set" given $x(0)$

3. implement and test $w_\mathrm{p} \leftrightarrow x(0)$ (`wp2x0` / `x02wp`)

Solution for part 1

$$y_{\mathrm{p}} = \underbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{\ell-1} \end{bmatrix}}_{\mathscr{O}} x(-\ell+1) + \underbrace{\begin{bmatrix} H(0) & & & \\ H(1) & H(0) & & \\ \vdots & \ddots & \ddots & \\ H(\ell-1) & \cdots & H(1) & H(0) \end{bmatrix}}_{\mathscr{T}} u_{\mathrm{p}}$$

$w_{\mathrm{p}} = \begin{bmatrix} u_{\mathrm{p}} \\ y_{\mathrm{p}} \end{bmatrix} \in \mathscr{B} \quad \Longrightarrow \quad$ solution $x(-\ell+1)$ exists

minimal repr. $\quad \Longrightarrow \quad \mathscr{O}$ full rank $\quad \Longrightarrow \quad x(-\ell+1)$ unique

$$x(0) = A^{\ell-1} x(-\ell+1) + \underbrace{\begin{bmatrix} A^{\ell-2}B & \cdots & BA^0 & 0 \end{bmatrix}}_{\mathscr{C}} u_{\mathrm{p}}$$

Solution for part 2 and 3

in order to set $x(0)$, we include a prefix $w_p \wedge w_f$

```
function x0 = wp2x0(wp, sys)
ell = size(sys, 'order');
xini  = obsv(sys) \ ...
        (wp(:, 2) - lsim(sys, wp(:, 1)));
C = [sys.b zeros(ell, 1)];
for i = 1:(ell - 2)
  C = [sys.a * C(:, 1) C];
end
x0 = sys.a ^ (ell - 1) * xini + C * wp(:, 1);
```

Solution for part 2 and 3

construct $\mathscr{C}$

```
C = [sys.b zeros(ell, 1)];
for i = 1:(ell - 2)
  C = [sys.a * C(:, 1) C];
end
```

construct $\mathscr{T}$

```
h = impulse(sys, ell - 1);
T = toeplitz(h, [h(1) zeros(1, ell - 1)]);
```

Solution for part 2 and 3 (continued)

$$x(0) = \begin{bmatrix} \mathscr{C} - A^{\ell-1}\mathscr{O}^{+}\mathscr{T} & A^{\ell-1}\mathscr{O}^{+} \end{bmatrix} \begin{bmatrix} u_{\rm p} \\ y_{\rm p} \end{bmatrix}$$

```
function wp = x02wp(x0, sys)
ell = size(sys, 'order');
<<construct-C>>
<<construct-T>>
O  = obsv(sys);
AO = sys.a ^ (ell -1) * pinv(O);
wp = pinv([C - AO * T , AO]) * x0;
wp = reshape(wp, ell, 2);
```

Solution (continued)

simulate data

```
n = 2; sys = drss(n);
T = 20; u = rand(T, 1); xini = rand(n, 1);
[y, t, x] = lsim(sys, u, [], xini); w = [u y];
```

test `wp2x0` and `x02wp`

```
<<simulate-data>>

wp = w(end - n + 1:end, :); x0 = x(end, :)';
wp2x0(wp, sys) - x0

wp2x0(x02wp(x0, sys), sys) - x0
```

Outline

Exact identification of a kernel representation

let $w \in \mathscr{B} \in \mathscr{L}^2_{1,\ell}$ (SISO system)

implement the method $w \mapsto R$ (slide 19)

test it on examples (use `drss`)

Solution

implementation

```
function r = w2r(w, ell)
r = null(blkhank(w, ell + 1)')';
```

test

```
<<simulate-data>>
sysh = r2tf(w2r(w, n));
norm(sys - sysh)
```

homework: generalize to the MIMO case

Impulse response estimation

let $w \in \mathscr{B} \in \mathscr{L}^2_{1,\ell}$ (SISO system)

implement the method $w \mapsto H$ (slide 20–21)

test it on examples (use `drss`)

Solution

implementation

```
function h = uy2h(u, y, ell, t)
L = ell + t;
H = [blkhank(u, L); blkhank(y, L)];
wini_uf = zeros(2 * ell + t, 1);
wini_uf(ell + 1) = 1;
h = H(2 * ell + t + 1:end, :) * ...
      pinv(H(1:(2 * ell + t), :)) * wini_uf;
```

test

```
<<simulate-data>>
t  = 5;
h  = impulse(sys, t - 1);
hh = uy2h(u, y, n, t);
norm(h - hh)
```

Outline

Misfit computation using image repr.

given

- data $w = \big(w(1), \dots, w(T)\big)$ and
- LTI system $\mathscr{B} = \text{image}\big(P(\sigma)\big)$

derive method for computing

$$\text{misfit}(w, \mathscr{B}) := \min_{\widehat{w} \in \mathscr{B}} \|w - \widehat{w}\|_2$$

i.e., find the orthogonal projection of w on $\mathscr{B}$

$w \overset{?}{\in} \text{image}\left(P(\sigma)\right)$

$\iff$ there is v, such that $w = P(\sigma)v$

$\iff$ there is v, such that for $t = 1, \dots, T$
$w(t) = P_0 v(t) + P_1 v(t+1) + \cdots + P_\ell v(t+\ell)$

$\iff$ there is solution v of the system

$$\begin{bmatrix} w(1) \\ w(2) \\ \vdots \\ w(T) \end{bmatrix} = \underbrace{\begin{bmatrix} P_0 & P_1 & \cdots & P_\ell & & & \\ & P_0 & P_1 & \cdots & P_\ell & & \\ & & \ddots & \ddots & & \ddots & \\ & & & P_0 & P_1 & \cdots & P_\ell \end{bmatrix}}_{\mathscr{M}_{T+\ell}(P)} \begin{bmatrix} v(1) \\ v(2) \\ \vdots \\ v(T+\ell) \end{bmatrix}$$

Solution

we showed that

$$\widehat{w} \in \ker\big(R(\sigma)\big) \quad \iff \quad \widehat{w} = \mathscr{M}_T(P)v, \text{ for some } v$$

then the misfit computation problem

$$\operatorname{misfit}(w, \mathscr{B}) := \min_{\widehat{w} \in \mathscr{B}} \|w - \widehat{w}\|$$

becomes

$$\text{minimize} \quad \text{over } v \quad \|w - \mathscr{M}_T(P)v\|$$

this is standard least-norm problem

projector on $\mathscr{B} = \text{image}(P)$

$$\Pi_{\text{image}(P)} := \mathscr{M}_T(P)\big(\mathscr{M}_T^\top(P)\mathscr{M}_T(P)\big)^{-1}\mathscr{M}_T^\top(P)$$

misfit

$$\text{misfit}(w,\mathscr{B}) := \sqrt{w^\top\big(I - \Pi_{\text{image}(P)}\big)w}$$

and optimal approximation

$$\widehat{w} = \Pi_{\text{image}(P)}w$$

homework: misfit computation with $\mathscr{B} = \ker\big(R(\sigma)\big)$

Misfit computation using I/S/O representation

given

- data $w = \big(w(1), \dots, w(T)\big)$ and
- LTI system $\mathscr{B} = \mathscr{B}(A, B, C, D)$

derive method for computing

$$\operatorname{misfit}(w, \mathscr{B}) := \min_{\widehat{w} \in \mathscr{B}} \| w - \widehat{w} \|_2$$

i.e., find the orthogonal projection of w on $\mathscr{B}$

$w \overset{?}{\in} \mathscr{B}(A,B,C,D)$

$$\mathscr{B}(A,B,C,D) = \{\,(u,y) \mid \sigma x = Ax + Bu,\; y = Cx + Du\,\}$$

$$(u_d, y_d) \in \mathscr{B}(A,B,C,D) \quad \Longleftrightarrow \quad \exists\, x_{\text{ini}} \in \mathbb{R}^n, \text{ such that}$$

$$y = \underbrace{\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{T-1} \end{bmatrix}}_{\mathscr{O}_T(A,C)} x_{\text{ini}} + \begin{bmatrix} D & & & & \\ CB & D & & & \\ CAB & CB & D & & \\ \vdots & \ddots & \ddots & \ddots & \\ CA^{T-1}B & \cdots & CAB & CB & D \end{bmatrix} u$$

Solution

we showed that

$$\widehat{w} \in \mathscr{B}(A, B, C, D) \iff \widehat{y} = \mathscr{O}_T(A, C)\widehat{x}_{\text{ini}} + \mathscr{T}_T(H)\widehat{u}$$

then the misfit computation problem

$$\min_{\widehat{x}_{\text{ini}}, \widehat{u}} \quad \left\| \begin{bmatrix} u_d \\ y_d \end{bmatrix} - \begin{bmatrix} 0 & I \\ \mathscr{O}_T(A, C) & \mathscr{T}_T(H) \end{bmatrix} \begin{bmatrix} \widehat{x}_{\text{ini}} \\ \widehat{u} \end{bmatrix} \right\|$$

exploiting the structure in the problem

$\rightsquigarrow$ EIV Kalman filter

Latency computation using kernel repr.

given

- data w and
- LTI system $\mathscr{B}_{\text{ext}} = \ker\big(R(\sigma)\big)$ $\qquad$ $(w_{\text{ext}} := \left[\begin{smallmatrix}\widehat{e}\\ w\end{smallmatrix}\right])$

find an algorithm for computing

$$\text{minimize} \quad \text{over } e \quad \|\widehat{e}\| \quad \text{subject to} \quad (\widehat{e}, w) \in \mathscr{B}_{\text{ext}}$$

Solution

partition $R = \begin{bmatrix} R_e & R_w \end{bmatrix}$ conformably with $w_{\text{ext}} = \left[\begin{smallmatrix} e \\ w \end{smallmatrix}\right]$

by analogy with the derivation on page 41, we have

$$\begin{bmatrix} e \\ w \end{bmatrix} \in \ker\big(R(\sigma)\big) \iff \begin{bmatrix} \mathscr{M}_T(R_e) & \mathscr{M}_T(R_w) \end{bmatrix} \begin{bmatrix} e \\ w \end{bmatrix} = 0$$

the latency computation problem is

$$\min_{e} \quad \|e\|_2 \quad \text{subject to} \quad \mathscr{M}_T(R_e)e = -\mathscr{M}_T(R_w)w$$

the solution is given by

$$\widehat{e} = -\underbrace{\big(\mathscr{M}_T(R_e)^\top \mathscr{M}_T(R_e)\big)^{-1} \mathscr{M}_T(R_e)^\top}_{\mathscr{M}_T(R_e)^+} \mathscr{M}_T(R_w)w$$

Outline

Software

mosaic-Hankel low-rank approximation

http://slra.github.io/software.html

```
[sysh,info,wh] = ident(w, m, ell, opt)
```

- `sysh` — I/S/O representation of the identified model
- `opt.sys0` — I/S/O repr. of initial approximation
- `opt.wini` — initial conditions
- `opt.exct` — exact variables
- `info.Rh` — parameter R of kernel repr.
- `info.M` — misfit

```
[M, wh, xini] = misfit(w, sysh, opt)
```

demo file

Variable permutation

verify that permutation of the variables doesn't change the optimal misfit

```
T = 100; n = 2; B0 = drss(n);
u = randn(T, 1); y = lsim(B0, u) + 0.001 * ranc
[B1, info1] = ident([u y], 1, n); disp(info1.M)
   2.9736e-05
[B2, info2] = ident([y u], 1, n); disp(info2.M)
   2.9736e-05
disp(norm(B1 - inv(B2)))
   5.8438e-12
```

Output error identification

verify that the results of `oe` and `ident` coincide

```
T = 100; n = 2; B0 = drss(n);
u = randn(T, 1); y = lsim(B0, u) + 0.001 * ranc
opt = oeOptions('InitialCondition', 'estimate')
B1 = oe(iddata(y, u), [n + 1 n 0], opt);
B2 = ident([u y], 1, n, struct('exct', 1));
norm(B1 - B2) / norm(B1)

ans =

   1.4760e-07
```