TSRT14: Sensor Fusion Lecture 2

— Estimation theory for nonlinear models

Gustaf Hendeby

gustaf.hendeby@liu.se

LINKÖPING UNIVERSITY

Le 2: estimation theory in nonlinear models

Whiteboard:

- Nonlinear models
- Nonlinear *weighted least squares* (NWLS)
- NWLS connection to *maximum likelihood* (ML) estimation
- *Nonlinear transform* (NLT) and methods

Slides:

- Details on sensor models and methods
- Examples
- Dedicated least squares methods

Summary Lecture 1

- Linear model on batch form:

$$\mathbf{y} = \mathbf{H}x + \mathbf{e}, \quad \operatorname{cov}(\mathbf{e}) = \mathbf{R}.$$

- WLS minimizes the loss function

$$V^{WLS}(x) = (\mathbf{y} - \mathbf{H}x)^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{H}x).$$

- WLS solution

$$\hat{x} = \left(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{y}, \quad P = \left(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\right)^{-1}.$$

- LS special case with $\mathbf{R} = I$ and gives larger P.
- The fusion formula for two independent estimates is

$$\mathsf{E}(\hat{x}_1) = \mathsf{E}(\hat{x}_2) = x, \quad \operatorname{cov}(\hat{x}_1) = P_1, \quad \operatorname{cov}(\hat{x}_2) = P_2 \Rightarrow$$
$$\hat{x} = P\left(P_1^{-1}\hat{x}_1 + P_2^{-1}\hat{x}_2\right), \quad P = \left(P_1^{-1} + P_2^{-1}\right)^{-1}.$$

- If the estimates are not independent, P is larger than indicated.

Estimation Theory for Nonlinear Models

Chapter 3 Overview

- Model $y_k = h_k(x) + e_k$
- Tool 1: Nonlinear least squares NLS
- NWLS minimization approaches
- Tool 2: Nonlinear transformations NLT
- NLT applied in a direct and an indirect approach
- Ranging sensor example
- Radar sensor example
- Partly linear models

NWLS Theory

Nonlinear model

$$\begin{aligned} y_k &= h_k(x) + e_k, & \text{cov}(e_k) &= R_k, \quad k = 1, \dots, N, \\ \mathbf{y} &= \mathbf{h}(x) + \mathbf{e}, & \text{cov}(\mathbf{e}) &= \mathbf{R}. \end{aligned}$$

The NWLS solution minimizes

$$\hat{x}^{\text{NWLS}} = \arg\min_x V^{\text{NWLS}}(x) = \arg\min_x \tfrac{1}{2} \sum_{k=1}^{N} \big(y_k - h_k(x)\big)^T R_k^{-1} \big(y_k - h_k(x)\big).$$

ML for Gaussian noise with parameter dependent covariance $R(x)$

$$\hat{x}^{\text{ML}} = \arg\min_x \big[V^{\text{NWLS}}(x) + \tfrac{1}{2} \textstyle\sum_k \log\det\big(R_k(x)\big)\big].$$

Minimization Approaches

- **Grid:** evaluate $V(x)$ for a set of grid points $x^{(i)}$ and minimize.
- **Linearization:** first order Taylor expansion

  $$\bar{y}_k = y_k - h_k(\bar{x}) + h_k'(\bar{x})\bar{x} = h_k'(\bar{x})x + e$$

  and apply the WLS method to this linear model.
- **Optimization:** basic idea, iterate linearization and WLS. Gauss-Newton falls into this category.
- **Second order Taylor expansion:** compensatation for mean and covariance in second order term possible in WLS.

Nonlinear Transformations (NLT)

Problem: given a nonlinear mapping

$$z = g(u)$$

of a Gaussian variable

$$u \sim \mathcal{N}(\mu_u, P_u),$$

how to approximate the output with a new Gaussian distribution

$$z \sim \mathcal{N}(\hat{z}, P_z).$$

Such approximations have two applications:

- Direct approach: apply NLT to $x = h^{-1}(y - e)$.
- Indirect approach: apply NLT to $y = h(x) + e$.

NLT: Taylor methods

- **TT1:** first order Taylor transformation (a.k.a. Gauss approximation formula)

$$u \sim \mathcal{N}\big(\mu_u, P_u\big) \to z \sim \mathcal{N}\Big(g(\mu_u), g'(\mu_u)P_u\big(g'(\mu_u)\big)^T\Big).$$

- **TT2:** second order Taylor transformation

$$\begin{aligned} &u \sim \mathcal{N}\big(\mu_u, P_u\big) \\ &\to z \sim \mathcal{N}\Big(g(\mu_u) + \tfrac{1}{2}[\operatorname{tr}(g_i''(\mu_u)P_u)]_i, \\ &\qquad g'(\mu_u)P_u\big(g'(\mu_u)\big)^T + \tfrac{1}{2}\Big[\operatorname{tr}(P_u g_i''(\mu_u)P_u g_j''(\mu_u))\Big]_{ij}\Big). \end{aligned}$$

NLT: sample methods

- **MCT:** Monte Carlo transformation

$$\begin{aligned} u^{(i)} &\sim p_u(u^{(i)}), \quad i = 1, \dots, N, \\ z^{(i)} &= g(u^{(i)}), \\ \mu_z &= \frac{1}{N}\sum_{i=1}^{N} z^{(i)}, \\ P_z &= \frac{1}{N-1}\sum_{i=1}^{N}\big(z^{(i)} - \mu_z\big)\big(z^{(i)} - \mu_z\big)^T. \end{aligned}$$

- **UT:** unscented transformation. Similar to MCT, but deterministic samples and other (non-intuitive) weights. Example comes later.

Direct Approach Using NLT

Two step approach:

Let $x = z$, $g(u) = h^{-1}(u)$ and $u = y - e$ in the general NLT.

1. NLT (TT1, TT2, MCT or UT) gives Gaussian approximation $\mathcal{N}(\hat{x}_k, P_k)$ for each sensor observation $x = g(y - e)$.
2. The fusion formula gives $\mathcal{N}(\hat{x}, P)$ (with no further approximation).

Indirect Approach Using NLT

General Bayesian approach to estimation:

1. Assume a prior of $x \sim \mathcal{N}(\hat{x}, P^{xx})$:
2. Form the stochastic vector in the NLT $z = g(u)$ notation

$$u = \begin{pmatrix} x \\ e \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \bar{x} \\ 0 \end{pmatrix}, \begin{pmatrix} P^{xx} & 0 \\ 0 & R \end{pmatrix}\right).$$

3. Apply a NLT (TT1, TT2, MCT, UT) to the mapping

$$z = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ h(x, e) \end{pmatrix} \approx \mathcal{N}\left(\begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}, \begin{pmatrix} P^{xx} & P^{xy} \\ P^{yx} & P^{yy} \end{pmatrix}\right).$$

4. Apply the formula (Lemma 7.1)

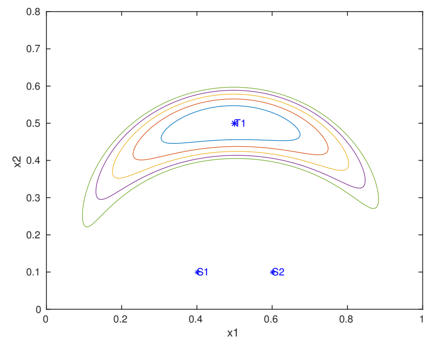
$$\begin{aligned} \hat{x} &= \bar{x} + P^{xy}(P^{yy})^{-1}(y - \bar{y}) \\ \operatorname{cov}(\hat{x}) &= P^{xx} - P^{xy}\big(P^{yy}\big)^{-1}P^{yx}. \end{aligned}$$

Trilateration

- Two sensors measure range only.
- Typical application: time of arrival (TOA) detection.
- Estimate is the intersection of two circles (trilateration).
- What is the uncertainty/precision/covariance?

NWLS for TOA (1/2)

```
th = [0.4, 0.1, 0.6, 0.1];  x0 = [0.5, 0.5];  % Positions
s = exsensor('toa', 2);                        % TOA sensor model
s.th = th;  s.x0 = x0;                         % Change defaults
s.pe = 0.00001 * eye(2);                       % Noise variance

y = simulate(s, 1);                            % Generate observations

plot(s); hold on                               % Plot network
lh2(s, y, 0:0.005:1, 0:0.005:.8);              % Likelihood function plot
```



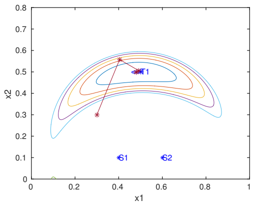
NWLS for TOA (2/2)

The likelihood function and the iterations in the NWLS estimate.

```
s0 = s;  s0.x0 = [0.3; 0.3];               % Prior model for estimation
[xhat, shat, res] = ml(s0, y);             % ML calls NWLS
shat                                       % Display result

plot(s);  hold on;                         % Plot network
lh2(s, y, 0:0.005:1, 0:0.005:.8);          % Likelihood function plot
plot2(shat.x0 + shat.px0, 'conf', 90,...
      'legend', 'off');                    % Estimate and covariance plot
plot(res.TH(1,:), res.TH(2,:), '*-')       % Estimate for each iteration
axis([0, 1, 0, .8]);
```

Output:

```
SENSORMOD object: TOA (calibrated from data)
    / sqrt((x(1,:)-th(1)).^2+(x(2,:)-th(2)).^2) \
 y = \ sqrt((x(1,:)-th(3)).^2+(x(2,:)-th(4)).^2) / + e
 x0' = [0.51,0.5]+N(0,[8.5e-05,-2.2e-06;-2.2e-06,5.4e-06])
 th' = [0.4,0.1,0.6,0.1]
 States:  x1      x2
 Outputs: y1      y2
```



Radar Observations

Sensor model:

$$y = (r, \varphi)^T + e = h(x_1, x_2) + e,$$
$$r = \sqrt{x_1^2 + x_2^2} + e_r,$$
$$\varphi = \arctan2(x_1, x_2) + e_\varphi.$$

Direct approach by inverting the observation model

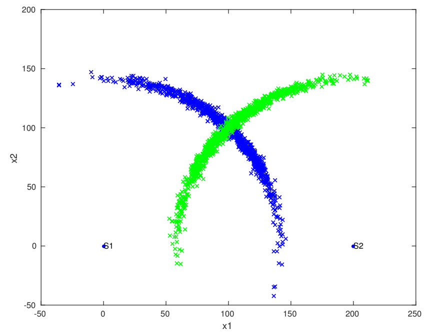
$$x = h^{-1}(y - e),$$
$$x_1 = y_1 \cos(y_2) = (r - e_r)\sin(\varphi - e_\varphi),$$
$$x_2 = y_1 \sin(y_2) = (r - e_r)\cos(\varphi - e_\varphi).$$

What is the covariance of $\hat{x} = h^{-1}(y)$?

Radar: Monte Carlo samples from direct method

- Generate measurements of range and bearing.
- Invert $x = h^{-1}(y)$ for each sample.
- Banana shaped distribution of estimates.

```
hinv= @(R, Phi, p) [p(1) + R * cos(Phi);
                    p(2) + R * sin(Phi)];

R1 = ndist(100 * sqrt(2), 5);
Phi1 = ndist(pi/4, 0.1);
p1 = [0; 0];

R2 = ndist(100 * sqrt(2), 5);
Phi2 = ndist(3 * pi/4, 0.1);
p2 = [200; 0];

xhat1 = hinv(R1, Phi1, p1);
xhat2 = hinv(R2, Phi2, p2);

plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.b',...
     'markersize', 15);
hold on;
text(p1(1), p1(2), 'S1');
text(p2(1), p2(2), 'S2');

plot2(xhat1, xhat2, 'legend', 'off');
```



Radar: analytic direct approach

Analytic approximation of cov(x).
For the radar sensor with $\hat{x} = h^{-1}(y)$, the covariance can be approximated with

$$\text{cov}(\hat{x}) = \frac{\sigma_r^2 - r^2\sigma_\varphi^2}{2}\begin{pmatrix} b + \cos(2\varphi) & \sin(2\varphi) \\ \sin(2\varphi) & b - \cos(2\varphi) \end{pmatrix}$$

$$b = \frac{\sigma_r^2 + r^2\sigma_\varphi^2}{\sigma_r^2 - r^2\sigma_\varphi^2}.$$

Approximation accurate if $r\sigma_\varphi^2/\sigma_r < 0.4$ and $\sigma_\varphi < 0.4$.
This is normally the case in radar applications.
It does not hold in the example where
$r\sigma_\varphi^2/\sigma_r = 100\sqrt{2} \cdot 0.1/\sqrt{5} \approx 6.3.$

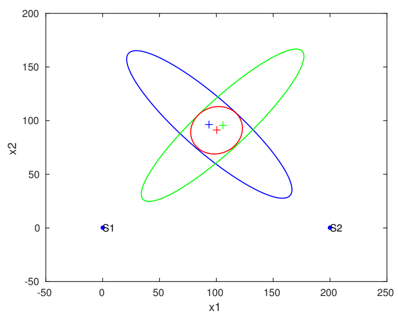
Radar: direct approach with MC

Fit a Gaussian to the Monte Carlo samples of y_k and apply the sensor fusion formula to the two Gaussian distributions.

```
V1 = [R1; Phi1]
Nhat1 = estimate(ndist, xhat1)
Nhat2 = estimate(ndist, xhat2)
xhat = fusion(Nhat1,Nhat2)

plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.b',...
     'markersize', 15);
hold on;
text(p1(1), p1(2), 'S1');
text(p2(1), p2(2), 'S2');
plot2(Nhat1, Nhat2, xhat, 'legend', 'off');
```

Output:

```
N([141;0.785],[5,0;0,0.1])
N([96.3;94.3],[947,-836;-836,905])
N([105;95.5],[940,852;852,937])
N([100;91],[85.4,1.34;1.34,83.4])
```



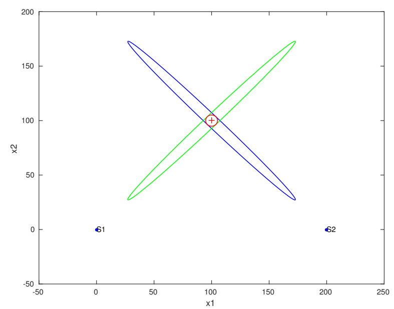
Radar: direct approach with TT1

Gauss approximation formula (based on linearizing $h_k^{-1}(x)$) applied to the banana transformation gives too optimistic result (since higher order terms are neglected).

```
y1 = [R1; Phi1];  y2 = [R2; Phi2];
hinv = @(y, p) [p(1) + y(1,:).*cos(y(2,:));
                p(2) + y(1,:).*sin(y(2,:))];

Nhat1 = tt1eval(y1, hinv, p1)
Nhat2 = tt1eval(y2, hinv, p2)
xhat = fusion(Nhat1, Nhat2)

plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.b',...
     'markersize', 15);
hold on;
text(p1(1), p1(2), 'S1');
text(p2(1), p2(2), 'S2');
plot2(Nhat1, Nhat2, xhat, 'legend', 'off');
```

Output:

```
N([100;100],[1e+03,-997;-997,1e+03])
N([100;100],[1e+03,998;998,1e+03])
N([100;100],[4.99,-2.18e-08;-2.18e-08,4.99])
```



Unscented Transformation

Method for transforming mean and covariance of y to $x = g(y)$:

1. Compute the *sigma points* $y^{(i)}$. These are the mean and symmetric deviations around the mean computed from the covariance matrix of y.
2. The sigma points are mapped to $x^{(i)} = h^{-1}(y^{(i)})$.
3. The mean and covariance are fitted to the mapped sigma points

$$\mu_x = \sum_{i=1}^{N} \omega_m^i x^{(i)},$$

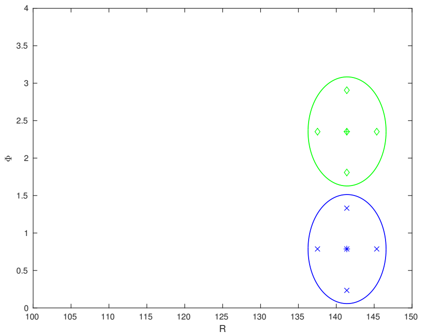
$$P_x = \sum_{i=1}^{N} \omega_c^i \big(x^{(i)} - \mu_x\big)\big(x^{(i)} - \mu_x\big)^T.$$

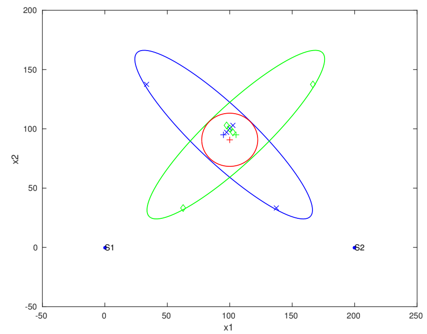
Tricks and rule of thumbs available to tune the weights.
Can be seen as a Monte Carlo method with deterministic sampling.

Radar: direct approach with UT (1/2)

Left: Distribution of $y = (r, \varphi)$ and sigma points $y^{(i)}$.

Right: Transformed sigma points $x^{(i)} = h^{-1}(y^{(i)})$ and fitted Gaussian distribution $\mathcal{N}(\hat{x}_k, P_k)$.





Blue: left sensor Green: right sensor Red: fused estimate

Radar: direct approach with UT (2/2)

```
[Nhat1, S1, fS1] = uteval(y1, hinv,...
                          'ut1', [], p1)
[Nhat2, S2, fS2] = uteval(y2, hinv,...
                          'ut1', [], p2)

plot2(y1, y2, 'legend', 'off');
hold on;
plot(S1(1, :), S1(2, :), 'xb',...
     S2(1, :), S2(2, :), 'dg');
```

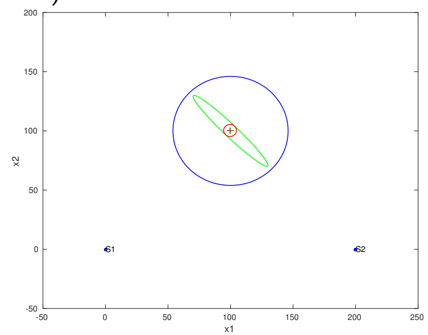
```
plot(p1(1), p1(2), '.b',...
     p2(1), p2(2), '.b',...
     'markersize', 15);
hold on;
text(p1(1), p1(2), 'S1');
text(p2(1), p2(2), 'S2');

plot2(Nhat1, Nhat2, xhat,...
      'legend', 'off');
hold on;
plot(fS1(1, :), fS1(2, :), 'xb',...
     fS2(1, :), fS2(2, :), 'dg');
```

Output:

```
N([95.1;95.1],[954,-854;-854,954])
S1 =
  141.4214  145.2943  141.4214  137.5484  141.4214
    0.7854    0.7854    1.3331    0.7854    0.2377
fS1 =
  100.0000  102.7386   33.2968   97.2614  137.4457
  100.0000  102.7386  137.4457   97.2614   33.2968
N([105;95.1],[954,854;854,954])
S2 =
  141.4214  145.2943  141.4214  137.5484  141.4214
    2.3562    2.3562    2.9039    2.3562    1.8085
fS2 =
  100.0000   97.2614   62.5543  102.7386  166.7032
  100.0000  102.7386   33.2968   97.2614  137.4457
N([100;90.8],[94.9,7.8e-14;-7.8e-14,94.9])
```


Radar: indirect approach with TT1 (1/2)

Gauss approximation formula (based on linearizing $h_k(x)$). The result is overly optimistic as higher order terms are neglected (*cf.* the direct TT1 approach).



Radar: indirect approach with TT1 (2/2)

```
g = @(x, p) [ x(1:2);                     % Function for joint distribution
              hypot(x(1)-p(1), x(2)-p(2)) + x(3);
              atan2(x(2)-p(2), x(1)-p(2)) + x(4); ];
Xhat0 = ndist([100; 100], 400*eye(2))  % Prior

e1 = ndist([0; 0], cov(y1));            % Approximate the joint distribtion
U1 = tt1eval([Xhat0; e1], g, p1);
[xhat1, P1] = condition(U1, mean(y1), [3, 4]);  % Perfrom conditioning
Xhat1 = ndist(xhat1, P1)

e2 = ndist([0; 0], cov(y2));
U2 = tt1eval([Xhat1; e2], g, p2);       % Approximate the joint distribtion
[xhat2, P2] = condition(U2, mean(y2), [3, 4]);  % Perfrom conditioning
Xhat2 = ndist(xhat2, P2)

plot(p1(1), p1(2), '.b',...
plot2(Xhat0, Xhat1, Xhat2, 'legend', off');
```

Output:

```
N([100;100],[400,0;0,400])
N([100;100],[169,-164;-164,169])
N([99.6;100],[4.93,0.0121;0.0121,4.93])
```


Conditionally Linear Models

$$y_k = h_k^n(x_n) + h_k^l(x_n)x_l + e_k, \quad \text{cov}(e_k) = R_k(x_n), \quad V^{\text{NWLS}}(x_n, x_l)$$

Separable least squares: The WLS solution for x_l is explicitly given by

$$\hat{x}_l^{\text{WLS}}(x_n) = \left(\sum_{k=1}^N \big(h_k^l(x_n)\big)^T R_k^{-1}(x_n) h_k^l(x_n)\right)^{-1} \sum_{k=1}^N \big(h_k^l(x_n)\big)^T R_k^{-1}(x_n)(y_k - h_k^n(x_n).$$

for each value of x_n. Which one to choose?

- Almost always utilize the separable least squares principle.
- In some cases, the loss function $\arg\min_{x_n} V^{\text{WLS}}\big(x_n, \hat{x}_l(x_n)\big)$ might have more local minima than the original formulation.

Summary

Summary Lecture 2 (1/2)

Nonlinear model

$$\mathbf{y} = \mathbf{h}(x) + \mathbf{e}, \quad \text{cov}(\mathbf{e}) = \mathbf{R}.$$

NWLS minimizes

$$\hat{x}^{\text{NWLS}} = \arg\min_x V^{\text{NWLS}}(x) = \arg\min_x \frac{1}{2}\big(\mathbf{y} - \mathbf{h}(x)\big)^T \mathbf{R}^{-1}\big(\mathbf{y} - \mathbf{h}(x)\big)$$

Linearization principle, replace $y_k = h_k(x) + e_k$ with $\bar{y}_k = y_k - h_k(\bar{x}) + h_k'(\bar{x})\bar{x} = h_k'(\bar{x})x + e$ and apply the WLS method to this linear model. Extensions:

- Iterate the linearization process.
- Compensate for the bias and variance of the rest term.

Model with linear sub-structure $h(x) = h^n(x^n) + h^l(x^n)x^l$. x^l can be eliminated with WLS, leading to a smaller search space.

Summary Lecture 2 (2/2)

NLT: Approximate $z = g(u)$, $u \sim \mathcal{N}(\hat{u}, P_u)$ with $z \sim \mathcal{N}(\hat{z}, P_z)$.
Variations: TT1, TT2, UT or MCT.

- The *direct approach*, where $x = \mathbf{h}^{-1}(\mathbf{y} - \mathbf{e})$ is approximated.
- The *indirect approach*, where the distribution of $\mathbf{y} = \mathbf{h}(x)$ is approximated using a prior of $x \sim \mathcal{N}(\hat{x}, P^{xx})$: The trick is to consider the mapping

$$u = \begin{pmatrix} x \\ e \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \bar{x} \\ 0 \end{pmatrix}, \begin{pmatrix} P^{xx} & 0 \\ 0 & R \end{pmatrix}\right)$$
$$z = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ h(x,e) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}, \begin{pmatrix} P^{xx} & P^{xy} \\ P^{yx} & P^{yy} \end{pmatrix}\right)$$

  and then apply

$$\hat{x} = \bar{x} + P^{xy}\big(P^{yy}\big)^{-1}(y - \bar{y}),$$
$$\mathrm{cov}(\hat{x}) = P^{xx} - P^{xy}\big(P^{yy}\big)^{-1}P^{yx}.$$