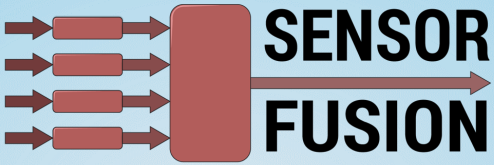


Nonlinear Transformations Using Samples

Sensor Fusion

Fredrik Gustafsson
fredrik.gustafsson@liu.se

Gustaf Hendeby
gustaf.hendeby@liu.se

Linköping University

Nonlinear Transformation (NLT) (of a stochastic variable)

In many cases it is important to perform nonlinear transformations of stochastic variables, e.g., for estimation of parameters with nonlinear measurement models.

Problem formulation: nonlinear transfomation (NLT)

Given the transform

$$z = g(u)$$

and the mean and covariance of the input,

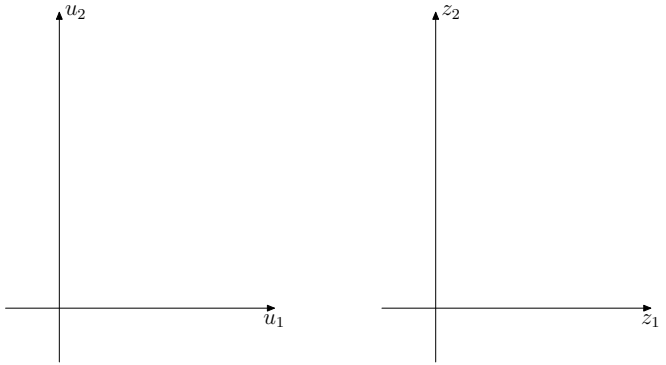
$$\mathrm{E}(u) = \mu_u, \quad \mathrm{Cov}(u) = P_u \quad \text{(often approximated } u \sim \mathcal{N}(\mu_u, P_u)\text{)}$$

determine

$$\mathrm{E}(z) = \mu_z \quad \mathrm{Cov}(z) = P_z \quad \text{(often approximated } z \sim \mathcal{N}(\mu_z, P_z)\text{)}.$$

Sample Based Transformation: idea

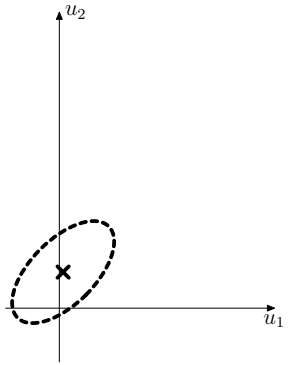
Select a number of samples from u and propagate them through the transformation and analyze the result to understand z.

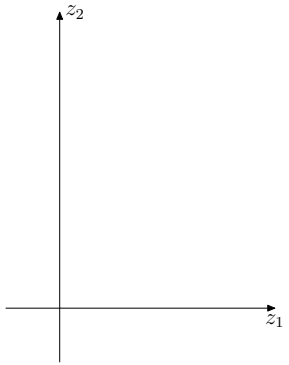


Sample Based Transformation: idea

Select a number of samples from u and propagate them through the transformation and analyze the result to understand z.

Initial distribution



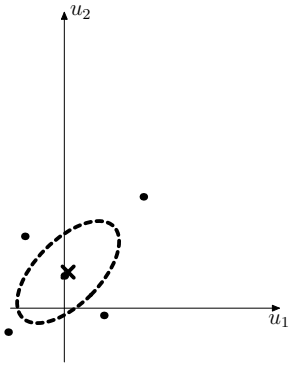


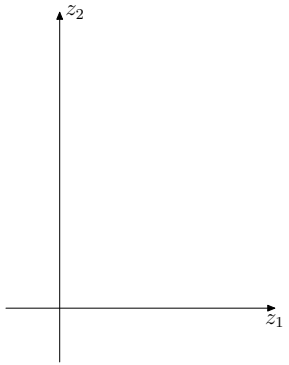
Sample Based Transformation: idea

Select a number of samples from u and propagate them through the transformation and analyze the result to understand z.

Initial distribution

1. Sample points



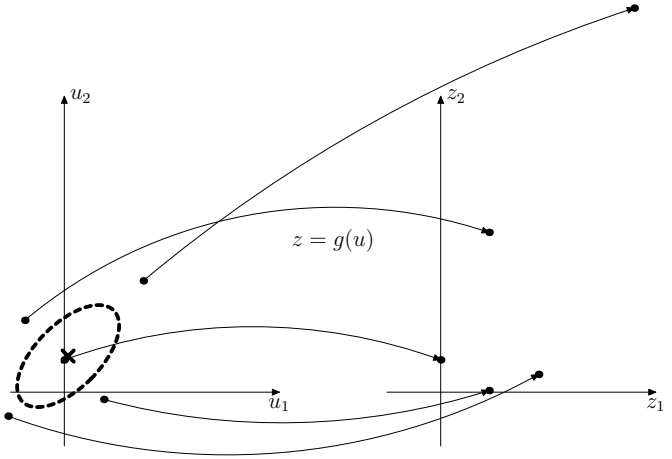


Sample Based Transformation: idea

Select a number of samples from u and propagate them through the transformation and analyze the result to understand z.

Initial distribution

1. Sample points
2. Transform points

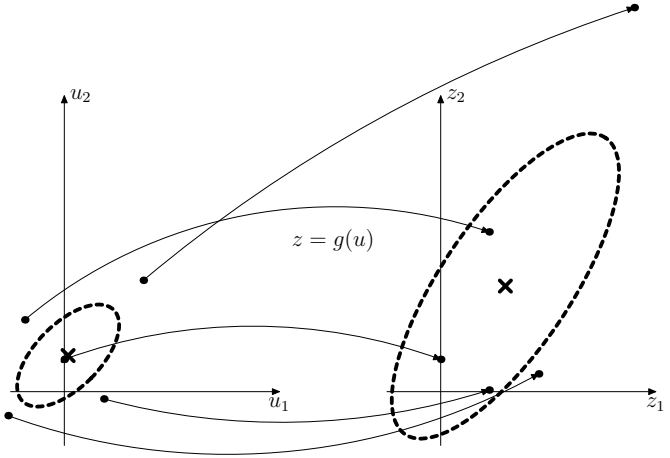


Sample Based Transformation: idea

Select a number of samples from u and propagate them through the transformation and analyze the result to understand z.

Initial distribution

1. Sample points
2. Transform points
3. Compute resulting moments



Monte Carlo Transformation (MCT)

In the Monte Carlo transformation a large number of independent random samples are drawn, and propagated through the nonlinear transformation ($i = 1, \ldots, N$):

$$u^{(i)} \sim p_u(u^{(i)}),$$
$$z^{(i)} = g(u^{(i)}).$$

The resulting approximation $\hat{x} \sim \mathcal{N}(\mu_z, P_z)$ follows as the sample mean and covariance:

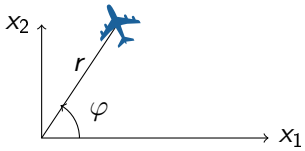
$$\mu_z = \frac{1}{N} \sum_{i=1}^{N} z^{(i)},$$

$$P_z = \frac{1}{N-1} \sum_{i=1}^{N} \left(z^{(i)} - \mu_z\right)\left(z^{(i)} - \mu_z\right)^T.$$

Radar Example: observations

Sensor model:

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



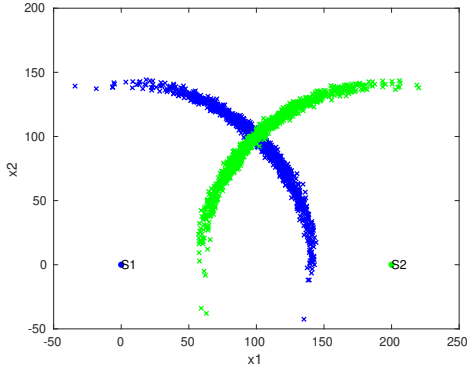
Direct approach, inverting the observation model

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

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

Radar Example: Monte Carlo samples

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



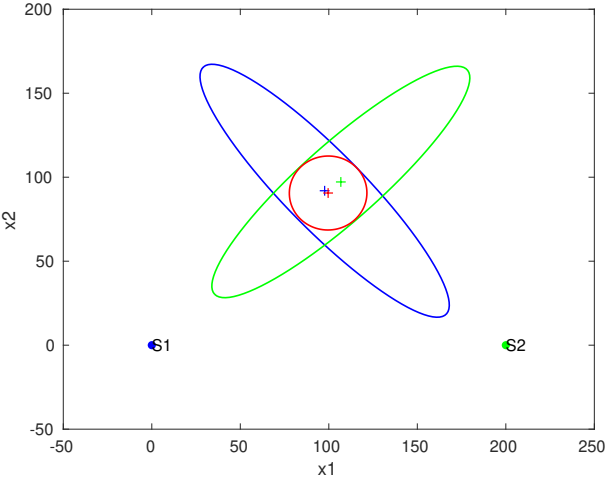
Matlab (SigSys):

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

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

Radar Example: MCT

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

```
Nhat1 = estimate(ndist, xhat1)
Nhat2 = estimate(ndist, xhat2)
xhat = fusion(Nhat1,Nhat2)

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



Output:

```
Nhat1 =
N([95;95.5],[946,-833;-833,929])
Nhat2 =
N([106;95.9],[1.03e+03,875;875,926])
xhat =
N([100;91],[99.8,1.79;1.79,93.8])
```

Unscented Transform (UT)

1. Generate $2n_u + 1$ *sigma points* and weights ($i = 1, 2, \ldots, n_u$):

$$u^{(0)} = \mu_u \qquad u^{(\pm i)} = \mu_u \pm \sqrt{n_u + \lambda}\left[P_u^{1/2}\right]_{:,i}$$

$$\omega^{(0)} = \frac{\lambda}{n_u + \lambda} \qquad \omega^{(\pm i)} = \frac{1}{2(n_u + \lambda)}$$

$$\omega_c^{(0)} = \omega^{(0)} + (1 - \alpha^2 + \beta) \qquad \omega_c^{(\pm i)} = \omega^{(\pm i)}$$

2. Transform the sigma points:

$$z^{(i)} = g(u^{(i)})$$

3. Compute the resulting mean and covariance:

$$\mu_z \approx \sum_{i=-n_u}^{n_u} \omega^{(i)} z^{(i)} \qquad P_z \approx \sum_{i=-n_u}^{n_u} \omega_c^{(i)} \left(z^{(i)} - \mu_z\right)\left(z^{(i)} - \mu_z\right)^T$$

Unscented Transform: parameters

- λ is defined by $\lambda = \alpha^2(n_u + \kappa) - n_u$.
- α controls the spread of the sigma points, suggested to be $\alpha \approx 10^{-3}$.
- β compensates for the distribution, and $\beta = 2$ for Gaussian distributions.
- κ is secondary scaling, usually $\kappa = 0$.

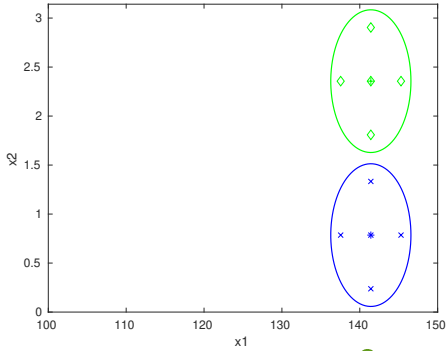
Note

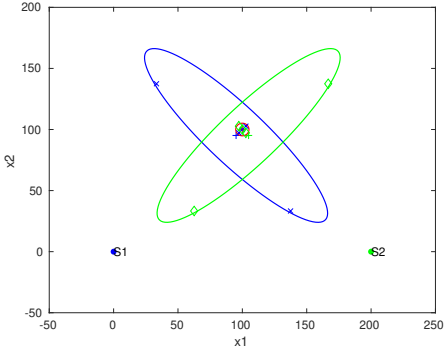
- $n_u + \lambda = \alpha^2 n_u$ when $\kappa = 0$.
- The weights sum to one for the mean, but sum to $2 - \alpha^2 + \beta \approx 4$ for the covariance. Note also that the weights are not necessarily in $[0, 1]$.
- The mean has a large negative weight!

Radar Example: direct UT (1/2)

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

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





Blue: left sensor

Green: right sensor

Red: fused estimate

Radar: direct approach with UT (2/2)

Matlab (SigSys):

```
[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), '.g',...
     '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:

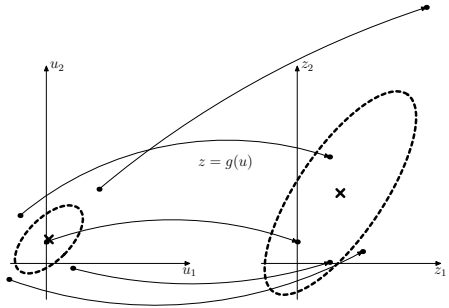
```
Nhat1 =
N([95.1;95.1],[954,-854;-854,954])
S1 =
  141.4214  145.2943  141.4214  137.5484  141.4214
    0.7854    0.7854    1.3331    0.7854    0.2377
fS1 =
  100.0000  102.7386   33.2968   97.2614  137.4457
  100.0000  102.7386  137.4457   97.2614   33.2968
Nhat2 =
N([105;95.1],[954,854;854,954])
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
```

Summary: sample based NLT

Nonlinear transformations (NLT) of stochastic variables are used to find the distribution of $z = g(u)$, given the mean and covariance of u.

Sample based methods: (1) Select a number of (random/deterministic) samples; (2) propagate them through the nonlinear transformation; and (3) analyze the result.

- Monte Carlo transformation, use a large number of random samples (particles).
- Unscented transformation, use a small set cleverly chosen samples (sigma points).



Sections 3.4.2 and 3.4.4