# TTK4250 – Sensor Fusion
## Assignment 1

Sigurd Totland | MTTK

September 4, 2019

## Task 1

Let the cdf of $Y$ be $P_Y(y)$. From its definition we obtain

$$P_Y(y) = \Pr(Y \leq y) = \Pr(P(X) \leq y).$$

We assume monotonicity of $P(X)$ which means that its inverse is defined. By this we obtain

$$\begin{aligned} P_Y(y) &= \Pr(P^-(P(X)) \leq P^-(y)) \\ &= \Pr(X \leq P^-(y)) \\ &= P(P^- y)) \\ &= y. \end{aligned}$$

Since the cdf of $Y$ is a unit ramp, $Y$ is uniformly distributed. ■

## Task 2

### a)

The probability generating function for a discrete random variable is defined as

$$G(t) = \sum_{x=0}^{\infty} f_X(x) t^x. \tag{1}$$

Using the power series definition $e^t = \sum_{x=0}^{\infty} \frac{t^x}{x!}$ and the probability mass function of the Poisson distribution $f_X(x) = \frac{\lambda^x e^{-\lambda}}{x!}$ we obtain

$$\begin{aligned} G(t) &= \sum_{x=0}^{\infty} \frac{\lambda^x e^{-\lambda}}{x!} t^x \\ &= e^{-\lambda} \sum_{x=0}^{\infty} \frac{(\lambda t)^x}{x!} \\ &= e^{-\lambda} e^{\lambda t} \\ &= e^{\lambda(t-1)}. \quad \blacksquare \end{aligned}$$

## b)

The pmf of the Binomial distribution is $f_X(x) = \binom{n}{x} r^x (1-r)^{n-x} t^x$. We then obtain the pgf

$$\begin{aligned} G(t) &= \sum_{x=0}^{n} f_X(x) t^x \\ &= \sum_{x=0}^{n} \binom{n}{x} r^x (1-r)^{n-x} t^x \\ &= \sum_{x=0}^{n} \frac{n!}{x!(n-x)!} (rt)^x (1-r)^{n-x} \\ &= [(1-r) + rt]^n, \end{aligned}$$

where the last result is obtained using the binomial theorem. ■

## c)

Letting $r = \frac{\lambda}{n}$ and taking the limit as $n \to \infty$ we obtain

$$\lim_{n\to\infty} [(1 - \frac{\lambda}{n}) + \frac{\lambda}{n} t]^n = (1 + \frac{\lambda(t-1)}{n})^n = e^{\lambda(t-1)}$$

which happens to be the pgf for the Poisson distribution. Informally speaking, this is because a Poisson distribution is just an infinite sequence of Bernoulli trials, so $n$ becomes infinite instead of finite.

## d)

**Using PGF**
For the sum of (in this case two) independent random variables, the pgf becomes

$$G_N(t) = G_{N_1}(t) G_{N_2}(t), \tag{2}$$

i.e. the product of the respective pgfs. For Poisson distributed variables $N_1$ and $N_2$ this becomes

$$G_N(t) = e^{\lambda_1(t-1)}e^{\lambda_2(t-1)} = e^{(\lambda_1+\lambda_2)(t-1)}. \tag{3}$$

N is thus Poisson distributed with probability parameter $\lambda$. ■

**Using PMF**

For the pmf we instead have

$$\begin{aligned} f_N(t) &= (f_{N_1} * f_{N_2})(t) \\ &= \sum_{x=0}^{t} \frac{\lambda_1^x e^{-\lambda_1}}{x!} \frac{\lambda_2^{t-x} e^{-\lambda_2}}{(t-x)!} \\ &= \frac{e^{-(\lambda_1+\lambda_2)}}{t!} \sum_{x=0}^{t} \frac{t!}{x!(t-x)!} \lambda_1^x \lambda_2^{t-x} \\ &= \frac{(\lambda_1+\lambda_2)^t e^{-(\lambda_1+\lambda_2)}}{t!} \\ &= \mathrm{Poisson}(t; \lambda_1+\lambda_2). \quad \blacksquare \end{aligned}$$

Clearly, using the probability generating function is much simpler than just the probability mass function. This is due to the fact that random variables addition becomes simple multiplication in the pgf case, but becomes a convolution operation in the pmf case which is generally much harder to compute.

# Task 3

## a)

We first rewrite the expression as

$$p(t_1|t_1 \geq t_0) = \Pr(T > t_1|T > t_0). \tag{4}$$

From Bayes' theorem we obtain

$$\Pr(T > t_1|T > t_0) = \frac{\Pr(T > t_0|T > t_1)\Pr(T > t_1)}{\Pr(T > t_0)}. \tag{5}$$

Assuming $t_1 > t_0$, if $T > t_1$, then $T > t_0$, so $\Pr(T > t_0|T > t_1) = 1$. We then get

$$\Pr(T > t_1|T > t_0) = \frac{\Pr(T > t_1)}{\Pr(T > t_0)}. \tag{6}$$

Using the pdf of the exponential distribution, we finally obtain

$$\Pr(T > t_1|T > t_0) = \frac{e^{-\lambda t_1}}{e^{-\lambda t_0}} = e^{-\lambda(t_1-t_0)}. \quad \blacksquare \tag{7}$$

This result is known as memorylessness. To interpret what this means intuitively, we must first how the exponential distribution is interpreted. In this problem, the probability density function $p(t)$ tells us the probability that *the next boat* will arrive in $t$ minutes. Memorylessness in this regard then means that even though we might have waited some time $t_0$ for a boat to arrive, the probability that it will come in $t_1 - t_0$ minutes does not change. It is useful for us, because this is how the world works when the boats are independent. (Had they not been, if they for instance worked together to keep their frequence of arrival as constant as possible, we would not have the memorylessness property.)

### b)

We use the result from example 2.9, i.e. that a sum of $n$ exponentially distributed variables are gamma distributed with scale parameter $\frac{1}{\lambda}$ and shape parameter $n$. Likewise, we obtain

$$t_n - t_0 = \sum_{i=1}^{n} t_i - t_{1-i} \sim \text{Gamma}(n, \frac{1}{\lambda}). \tag{8}$$

### c)

Due to the memorylessness property, this is the same as asking what the probability is that boat $n+1$ arrives after $T - t_n$ time from now, i.e.

$$\Pr(t_{n+1} > T - t_n) = 1 - \Pr(t_{n+1} \leq T - t_n) \tag{9}$$

Since the CDF of the exponential function is $1-e^{-\lambda t}$, we get that this probability equals

$$1 - \Pr(t_{n+1} \leq T - t_n) = 1 - 1 + e^{-\lambda(T-t_n)} = e^{-\lambda(T-t_n)}. \tag{10}$$

## Task 4

### a)

The moment generating function of $z$ is

$$M_z(t) = \text{E}[e^{t^\top \Sigma^{-\frac{1}{2}}}(x-\mu)] = e^{-t^\top \Sigma^{-\frac{1}{2}}\mu}\text{E}[e^{t^\top \Sigma^{-\frac{1}{2}}x}]. \tag{11}$$

We notice that $\text{E}[e^{t^\top \Sigma^{-\frac{1}{2}}x}]$ is the moment generating function of the random variable $\Sigma^{-\frac{1}{2}x}$. By linearity of the multivariate Gaussian, we know that

$$\Sigma^{-\frac{1}{2}}x \sim \mathcal{N}(\Sigma^{-\frac{1}{2}}\mu, \Sigma^{-\frac{1}{2}}\Sigma(\Sigma^{-\frac{1}{2}})^\top). \tag{12}$$

Furthermore, using the property $\Sigma^{\frac{1}{2}}(\Sigma^{\frac{1}{2}})^\top = \Sigma$ we have

$$\Sigma^{-\frac{1}{2}}\Sigma(\Sigma^{-\frac{1}{2}}) = \Sigma^{-\frac{1}{2}}\Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}}(\Sigma^{-\frac{1}{2}}) = \text{I}. \tag{13}$$

We also know that the moment generating function of any multivariate gaussian with expectation $\mu_0$ and variance $\Sigma_0$ is $M(t) = e^{t^\top(\mu_0 + \frac{1}{2}\Sigma_0 t)}$. Hence,

$$M_z(t) = e^{-t^\top \Sigma^{-\frac{1}{2}}\mu} e^{t^\top \Sigma^{-\frac{1}{2}}\mu} e^{\frac{1}{2}t^\top \mathrm{I} t}. \tag{14}$$

From this, we deduce that $z \sim \mathcal{N}(0, \mathrm{I})$, i.e. $z$ is standard normal distributed.

## b)

Since $z_i \sim \mathcal{N}(0, 1)$, this problem closely follows example 2.11. Since the square function is not invertible, we need to define both

$$\mathbf{f}_1^{-1} = \sqrt{y_i} \quad \text{and} \quad \mathbf{f}_2^{-1} = -\sqrt{y_i}. \tag{15}$$

Their jacobians are

$$\mathbf{F}_1^{-1} = \frac{1}{2\sqrt{y_i}} \quad \text{and} \quad \mathbf{F}_2^{-1} = -\frac{1}{2\sqrt{y_i}}. \tag{16}$$

By applying theorem 2.4.1, we arrive at the pdf

$$h(y_i) = \frac{1}{\sqrt{2\pi}}\left[\frac{1}{2\sqrt{y_i}} e^{-\frac{1}{2}\sqrt{y_i}^2} e^{-\frac{1}{2}(-\sqrt{y_i}^2)}\right] = \frac{1}{\sqrt{2\pi y_i}} e^{-\frac{y_i}{2}}, \tag{17}$$

just like in the example. This is the pdf of a $\chi^2$ distribution with 1 degree of freedom.

## c)

From the previous task, we know that $y_i$ is $\chi^2$ with $k = 1$ degree of freedom. The MGF of any $\chi^2$ variable is $(1 - 2t)^{-\frac{1}{2}}$, so in this case $M_{y_i}(t) = (1 - 2t)^{-\frac{1}{2}}$. The MGF of $y = \sum_i y_i$ is then

$$\begin{aligned} M_y(t) &= \mathrm{E}[e^{t\sum_i y_i}] \\ &= \prod_i \mathrm{E}[e^{ty_i}] \\ &= \prod_i (1 - 2t)^{-\frac{1}{2}} \\ &= (1 - 2t)^{-\frac{n}{2}} \end{aligned} \tag{18}$$

where $n$ is the number of random variables in $y$. In other words, $y$, the *Mahalanobis distance* from $x$ to $\mu$, is $\chi^2$ distributed with $n$ degrees of freedom.

# Task 5

## a)

We know that $p(z^c)$ is the pdf of $z^c$ and $p(z^c|x)$ is simply the pdf of $z^c$ *given* the state $x$. From equation (4.71) in the book, we know this distribution is $\mathcal{N}(z^c; H\bar{x}, R)$, i.e. $p(z^c|x) = \mathcal{N}(z^c; H\bar{x}, R)$.

## b)

This result follows from the proof of theorem 3.3.1 in the book. Like in the proof, we have that $p(z^c|x) = \mathcal{N}(z^c; Hx, R)$, and in addition, $x \sim \mathcal{N}(x; \bar{x}, P)$. As such, $p(x, z^c) = p(z^c|x)p(x) = \mathcal{N}(z^c; Hx, R)\mathcal{N}(x; \bar{x}, P)$. The proof then goes on in equation (3.13) to rewrite the quadratic form of this distribution to

$$\begin{aligned} &(x - \hat{x})^\top P^{-1}(x - \bar{x}) + (z^c - H^c x)^\top (R^c)^{-1}(z^c - H^c x) \\ &= \begin{bmatrix} x - \bar{x} \\ z^c - H^c\bar{x} \end{bmatrix}^\top \begin{bmatrix} P & P(H^c)^\top \\ H^cP & H^cP(H^c)^\top + R^c \end{bmatrix}^{-1} \begin{bmatrix} x - \bar{x} \\ z^c - H^c\bar{x} \end{bmatrix} \end{aligned} \tag{19}$$

which we (and the proof) recognize as the quadratic form of a Gaussian. In other words, $p(z^c, x)$ is indeed Gaussian. ■

## c)

We know that $\mathrm{E}[z^c] = \mathrm{E}[Hx + v] = H\mathrm{E}[x] + \mathrm{E}[v] = H\bar{x}$. From theorem 3.2.3, with

$$\begin{bmatrix} P_{xx} & P_{xy} \\ P_{xy}^\top & P_{yy} \end{bmatrix} = \begin{bmatrix} P & P(H^c)^\top \\ H^cP & H^cP(H^c)^\top + R^c \end{bmatrix} \tag{20}$$

we then have that

$$p(z^c) = \mathcal{N}(z^c; \mu_{z^c}, P_{yy}) = \mathcal{N}(z^c; H\bar{x}, H^cP(H^c)^\top + R^c). \tag{21}$$

Furthermore, the theorem tells us that

$$\begin{aligned} \mu_{x|z^c} &= \bar{x} + P_{xx}P_{yy}^{-1}(z^c - \mu_{z^c}) \\ &= \bar{x} + P(H^cP(H^c)^\top + R^c)^{-1}(z^c - \bar{x}) \end{aligned} \tag{22}$$

and

$$\begin{aligned} P_{x|z^c} &= P_{xx} - P_{xy}P_{yy}^{-1}P_{xy}^\top \\ &= P - H^cP(H^cP(H^c)^\top + R^c)^{-1}P(H^c)^\top \end{aligned} \tag{23}$$

and then the conditional

$$\begin{aligned} p(x|z^c) &= \mathcal{N}(x; \mu_{x|z^c}, P_{x|z^c}) \\ &= \mathcal{N}(x; \bar{x} + P(H^cP(H^c)^\top + R^c)^{-1}(z^c - \mu_{z^c}), \\ &\quad P - H^cP(H^cP(H^c)^\top + R^c)^{-1}P(H^c)^\top). \end{aligned} \tag{24}$$

## d)

$F$ is not specified to be non-singular, which is required to be able to apply theorem 3.2.3. (In the proof, we need to take the inverse of $F$). If it *is* non-singular, it should probably be possible to derive the same result, just replacing $H$ with $F$ and $R^c$ with $Q$ to obtain

$$p(x^+) = \mathcal{N}(x^+; F\bar{x}, F^cP(F^c)^\top + Q^c). \tag{25}$$

For $p(x|z^r)$ we get the corresponding result,

$$\begin{aligned} p(x|z^r) &= \mathcal{N}(x; \mu_{x|z^r}, P_{x|z^r}) \\ &= \mathcal{N}(x; \bar{x} + P(H^r P(H^r)^\top + R^r)^{-1}(z^r - \mu_{z^r}), \\ P &- H^r P(H^r P(H^r)^\top + R^r)^{-1} P(H^r)^\top). \end{aligned} \tag{26}$$

## e)

From theorem 2.6.1, we know that the MMSE estimator is pretty much just the expected value, i.e.

$$\hat{x}_{\text{MMSE}} = \mu_{x|z^c} = \bar{x} + P_{xx} P_{yy}^{-1}(z^c - \mu_{z^c}). \tag{27}$$

The MAP estimator on the other hand is given by

$$\hat{x}_{\text{MAP}} = \arg\max_x p(x|z^c). \tag{28}$$

According to the book, the MAP estimator has a closed form solution when estimating the expected value of a Gaussian. For more general systems however, we must resort to search techniques like Newton's method or steepest descent.

## f) and g)

```
% initialize the values
x_bar = zeros(2,1)
```

```
x_bar =
     0
     0
```

```
P = 25 * eye(2)
```

```
P =
    25     0
     0    25
```

```

H_r = eye(2)
```

```
H_r =
     1     0
     0     1
```

```
H_c = eye(2)
```

```
H_c =
     1     0
     0     1
```

```

R_c = [79, 36; 36, 36]
```

```
R_c =
    79    36
    36    36
```

```
R_r = [28, 4; 4, 22]
```

```
R_r =
    28     4
     4    22
```

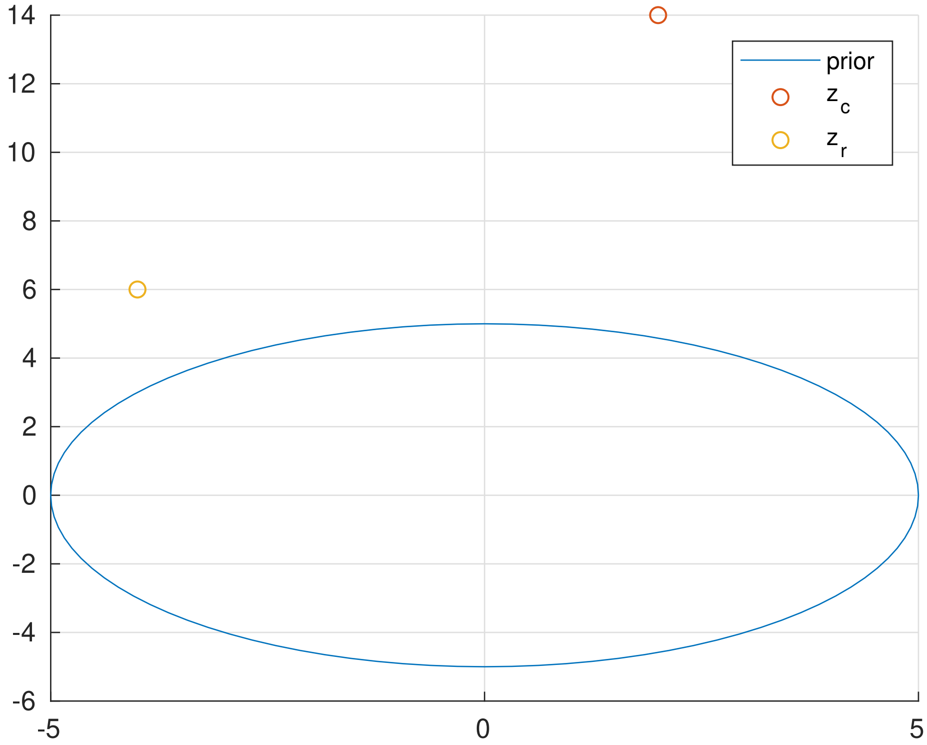
```

z_c = [2; 14]
```

```
z_c =
     2
    14
```

```
z_r = [-4; 6]
```

```
z_r =
    -4
     6
```

```


% set up for plotting ellipses
Npts = 100; % number of points around the circle
circle = [cos( 0:(2*pi/Npts):(2*pi) ); sin( ...
    0:(2*pi/Npts):(2*pi) )];

% initial 1 sigma ellipses
figure(1); clf; hold on; grid on;
data = x_bar + chol(P)' * circle;
plot(data(1,:), data(2, :), 'DisplayName','prior')

% measurements
scatter(z_c(1), z_c(2), 'DisplayName', 'z_c')
scatter(z_r(1), z_r(2), 'DisplayName', 'z_r')

legend()
```



```
% You can make some functions to ease the ...
    conditionioning
% FILL IN THE DOTS ...
condition_mean = @(x_bar, z, P, H, R) x_bar + ...
    P*(H*P*H' + R)^(-1)*(z-x_bar);
condition_cov = @(P, H, R) P - H*P*(H*P*H' + ...
    R)^(-1)*P*H';
```

```
% task 5 (f)
% FILL IN FOR THE DOTS ...

% condition on camera
x_bar_c = condition_mean(x_bar, z_c, P, H_c, ...
    R_c)
```

```
x_bar_c =
   -1.8918
    6.8542
```

```
P_c = condition_cov(P, H_c, R_c)
P_c =
   17.4475    4.4572
    4.4572   12.1236
```

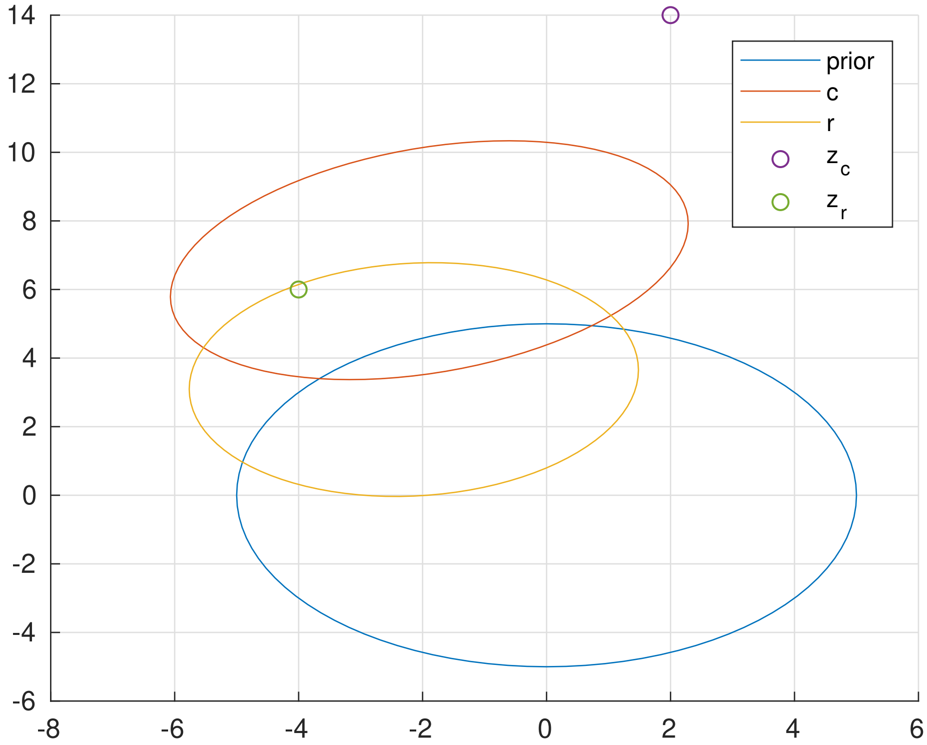
```

% condition on radar
x_bar_r = condition_mean(x_bar, z_r, P, H_r, ...
    R_r)
```

```
x_bar_r =
   -2.1414
    3.3737
```

```
P_r = condition_cov(P, H_r, R_r)
```

```
P_r =
   13.1313    1.0101
    1.0101   11.6162
```

```

% Plot 1 sigma ellipses
figure(2); clf; hold on; grid on;

data = x_bar + chol(P)' * circle;
plot(data(1,:), data(2, :), 'DisplayName','prior')

data = x_bar_c + chol(P_c)' * circle;
plot(data(1,:), data(2, :), 'DisplayName', 'c')

data = x_bar_r + chol(P_r)' * circle;
plot(data(1,:), data(2, :), 'DisplayName', 'r')

% measurements
scatter(z_c(1), z_c(2), 'DisplayName', 'z_c')
scatter(z_r(1), z_r(2), 'DisplayName', 'z_r')
legend()
```



```
% task 5 (g)

% condition the already camera conditioned on the ...
   radar
x_bar_cr = condition_mean(x_bar_c, z_r, P_c, ...
   H_r, R_r)
```

```
x_bar_cr =
   -2.7183
    6.4872
```

```
P_cr =  condition_cov(P_c, H_r, R_r)
```

```
P_cr =
   10.7043    2.3261
    2.3261    7.7676
```

```

% condition the already radar conditioned on the ...
   camera
```

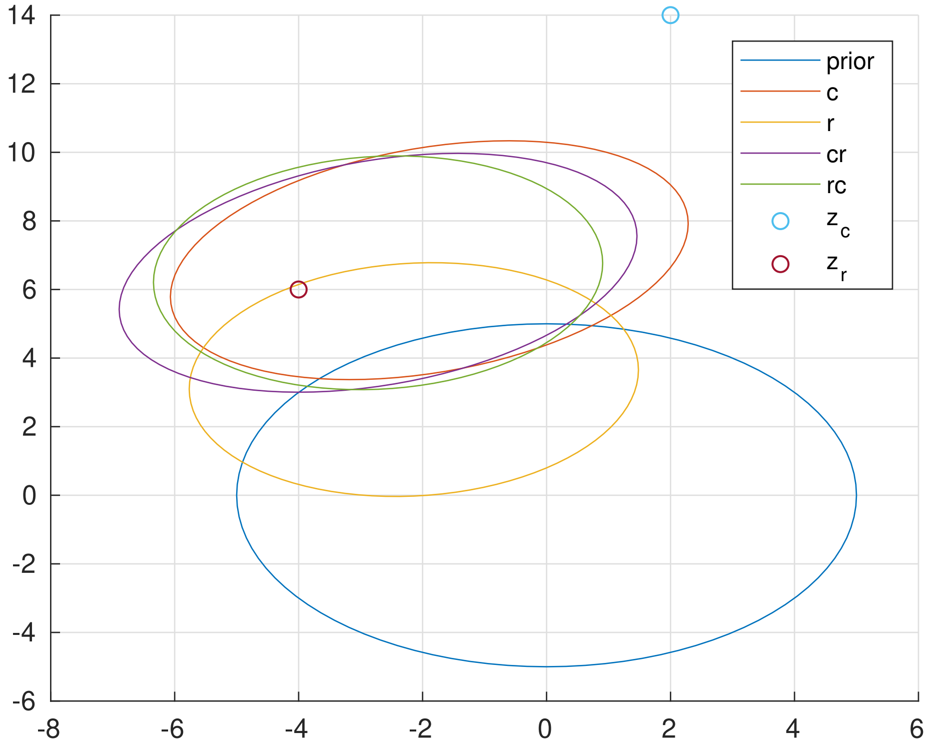
```
x_bar_rc = condition_mean(x_bar_r, z_c, P_r, ...
    H_c, R_c)
```

```
x_bar_rc =
    -2.7183
     6.4872
```

```
P_rc = condition_cov(P_r, H_c, R_c)
```

```
P_rc =
    10.7043    2.3261
     2.3261    7.7676
```

```


% Plot 1 sigma ellipses
figure(3); clf; hold on; grid on;

data = x_bar + chol(P)' * circle;
plot(data(1,:), data(2, :), 'DisplayName','prior')

data = x_bar_c + chol(P_c)' * circle;
plot(data(1,:), data(2, :), 'DisplayName', 'c')

data = x_bar_r + chol(P_r)' * circle;
plot(data(1,:), data(2, :), 'DisplayName', 'r')

data = x_bar_cr + chol(P_c)' * circle;
plot(data(1,:), data(2, :), 'DisplayName', 'cr')

data = x_bar_rc + chol(P_r)' * circle;
plot(data(1,:), data(2, :), 'DisplayName', 'rc')

% meausrements
scatter(z_c(1), z_c(2), 'DisplayName', 'z_c')
scatter(z_r(1), z_r(2), 'DisplayName', 'z_r')

legend()
```


14
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
prior
c
r
cr
rc
$z_c$
$z_r$

As we can see, it *does* matter in which order we condition, as the distributions are not the same in the resulting plot.

### h)

As we can see from figure 1, big prob.

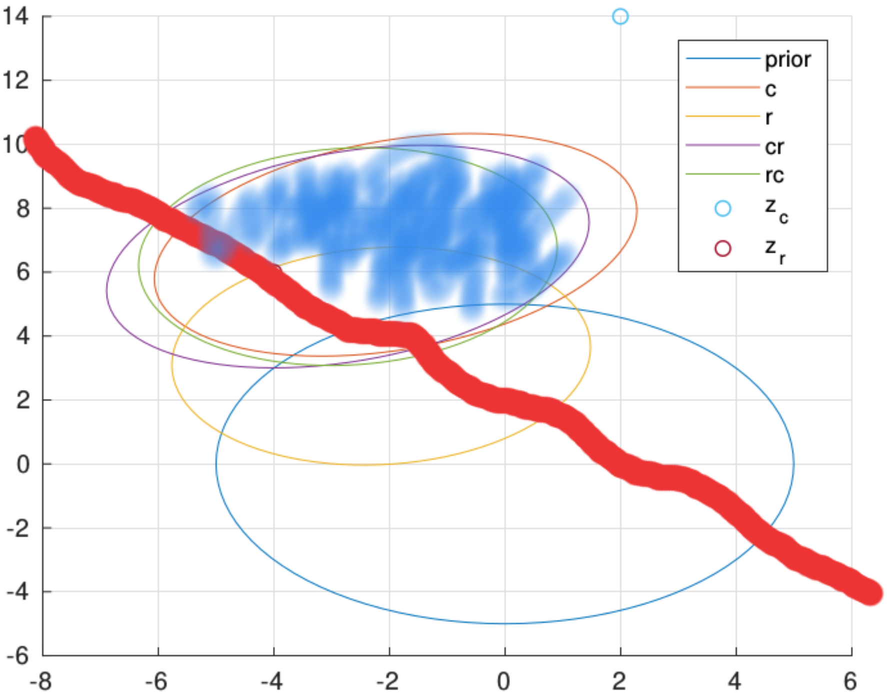


Figure 1: Big prob.