ENSTA Bretagne

Observateurs et filtre de Kalman

Fabrice LE BARS

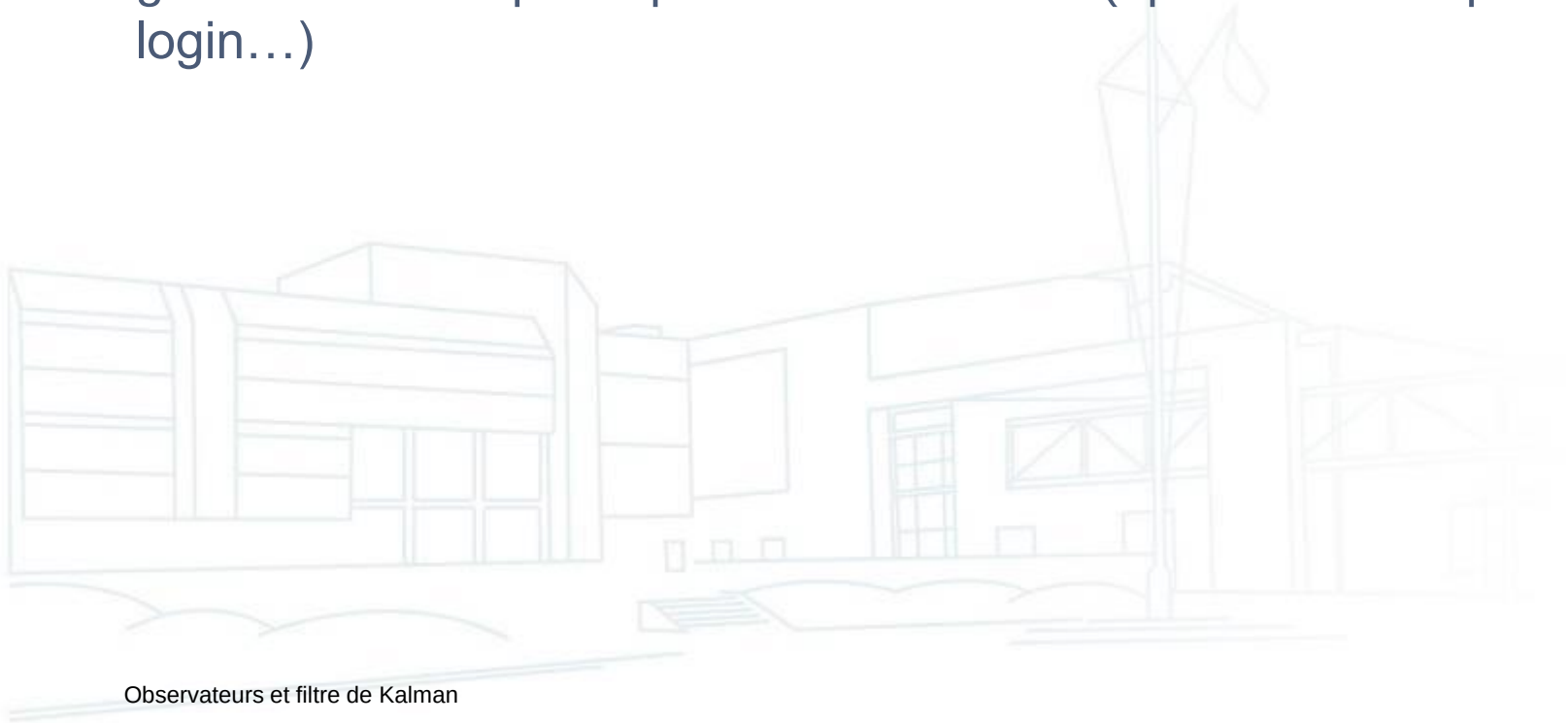
Présentation générale de la matière

Présentation générale de la matière

- Objectifs
  - Etre sensibilisé aux problèmes d’estimation d’état
  - Utiliser différents types d'observateurs d'état
  - Savoir implémenter un filtre de Kalman dans des cas simples
  - Avoir des notions sur ses applications et ses variantes courantes
  - Connaitre le vocabulaire courant autour de la notion de filtre de Kalman

Présentation générale de la matière

- Planning
  - 30C
  - Majorité des travaux sur ordinateur (MATLAB)
  - Se mettre à 1 par PC en salle info, sous Windows et essayer de garder la même place pour tous les cours (optimise le temps de login…)



Présentation générale de la matière

- Planning
  - Révisions et travaux préliminaires (2x4C)
  - Vocabulaire et notions prérequises : vecteurs aléatoires, bruits, gaussiennes, matrices de covariance, etc. (1x4C)
  - Formalisation du filtre de Kalman (2x4C?)
  - Filtre de Kalman et régulation (2x4C?)
  - Extended Kalman Filter (?)
  - Lisseur de Kalman (?)
  - Filtres particulaires, ensemblistes ou autres techniques (?)
  - TE de révision (?)
  - Evaluation (2C)

Révisions et travaux préliminaires

Révisions et travaux préliminaires

- Révisions
  - Cours d'automatique de 1A :

    Certaines notions directement réutilisées (**équations d'état**, **simulation** avec Euler…),

    D'autres en versions différentes (**observateurs** (en 1A ils étaient surtout pour les **systèmes linéaires invariants**), **régulateurs**…)

Révisions et travaux préliminaires

- Programme
  - Modélisation de systèmes avec des équations d'état
  - Simulation par méthode d'Euler
  - Coordonnées homogènes
  - Observateur d'état
  - Fusion de données
  - Commande
  - Implémentation en MATLAB : en simulation, et avec un système presque réel

Révisions et travaux préliminaires

- Modélisation de systèmes avec des équations d'état
  - Le fonctionnement de très nombreux systèmes (voiture, bateau, processus chimique, économique…) peut être modélisé par des équations d'état
  - Equation d'état/représentation d'état :

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & \text{équation d'évolution} \\ \mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)) & \text{équation d'observation} \end{cases}$$

  - Variables d'état : en général les variables nécessaires pour dessiner le système à un t donné + celles permettant de prévoir ce qui se passera au t suivant

    A noter qu'il faut aussi qu'on puisse écrire leur dérivée et qu'on ne les ait pas déjà via une relation non-différentielle avec les entrées (dans ce cas ce n'est qu'une variable intermédiaire)

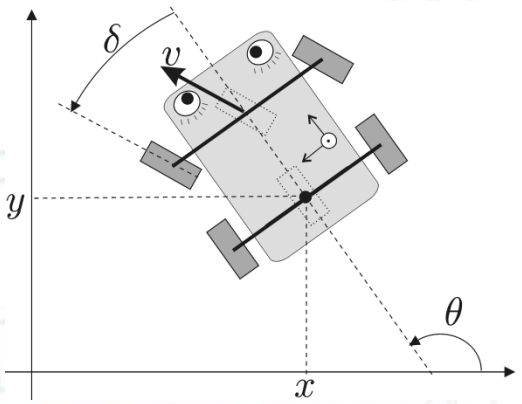
Révisions et travaux préliminaires

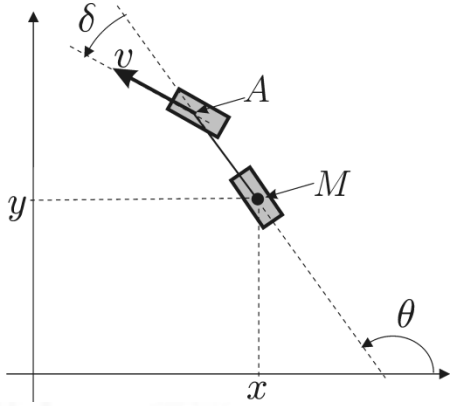
- Modélisation de systèmes avec des équations d'état
  - **Etat** : vecteur souvent noté $\mathbf{x}$ regroupant les variables d'état
  - **Entrées** : vecteur souvent noté $\mathbf{u}$ regroupant en général les signaux de commande directement envoyés au système, ou parfois leurs mesures plus ou moins directes
  - **Sorties** : vecteur souvent noté $\mathbf{y}$ regroupant en général les variables intéressantes mesurées par les capteurs du système

Révisions et travaux préliminaires

- Modélisation de systèmes avec des équations d'état
  - **Equation d'évolution** : **équation différentielle** permettant de savoir vers où va se diriger l'état x(t) sachant sa valeur à l'instant présent t et la commande u(t) actuelle
  - **Equation d'observation** : **calcul des sorties** y(t) actuelles en fonction de l'état actuel x(t) et la commande actuelle u(t)
  - Exemple d'équation d'évolution : voiture simplifiée

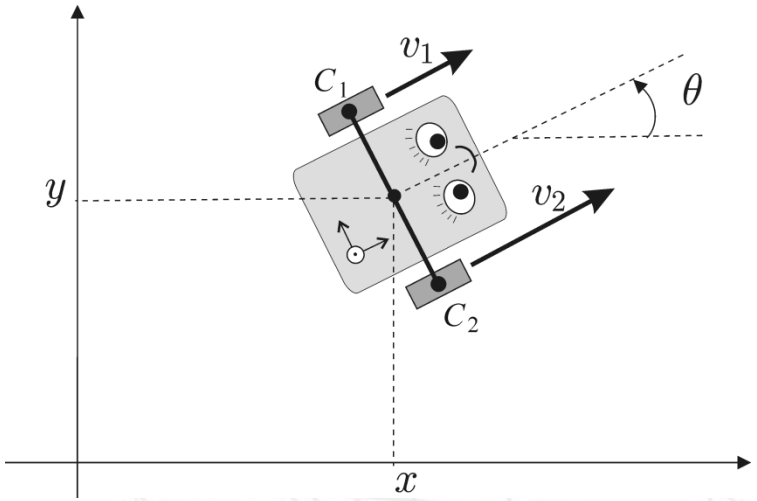
$$\underbrace{\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\delta} \end{pmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{pmatrix} v\cos\delta\cos\theta \\ v\cos\delta\sin\theta \\ \frac{v\sin\delta}{L} \\ u_1 \\ u_2 \end{pmatrix}}_{\mathbf{f}(\mathbf{x},\mathbf{u})}$$





Révisions et travaux préliminaires

- Modélisation de systèmes avec des équations d'état
  - Exemple : char



$$\begin{cases} \dot{x} = \frac{r\alpha_1 u_1 + r\alpha_2 u_2}{2}\cos\theta \\ \dot{y} = \frac{r\alpha_1 u_1 + r\alpha_2 u_2}{2}\sin\theta \\ \dot{\theta} = \frac{r\alpha_2 u_2 - r\alpha_1 u_1}{l} \end{cases}$$

$$v = \frac{v_1+v_2}{2}$$

$$\omega = \frac{v_2-v_1}{l}$$

$$\omega_1 = \alpha_1 u_1$$

$$\omega_2 = \alpha_2 u_2$$

$$v_1 = r\omega_1$$

$$v_2 = r\omega_2$$

r Rayon des roues

l Distance entre roues

Révisions et travaux préliminaires

- Modélisation de systèmes avec des équations d'état
  - Exemple : autre type de char

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v\cos\theta \\ v\sin\theta \\ u_1 \\ u_2 \end{pmatrix}$$

  - Exemple : modèle de robot simple et assez général évoluant en 2.5D (e.g. quadrirotor, sous-marin…), souvent utilisé en post-traitement

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{pmatrix} u_1\cos u_4 - u_2\sin u_4 \\ u_1\sin u_4 + u_2\cos u_4 \\ u_3 \end{pmatrix}$$

Révisions et travaux préliminaires

- Simulation par méthode d'Euler
  - Une fois qu'on a trouvé des équations d'état pour un système, il est bon de faire une simulation pour voir si elles représentent bien son comportement
  - Vu que l'équation d'évolution est une **équation différentielle**, on peut utiliser une **méthode d'intégration numérique** comme la **méthode d'Euler** :

Avec $\mathbf{x}(t+dt) \simeq \mathbf{x}(t) + dt.\dot{\mathbf{x}}(t)$

Vu l'équation d'évolution $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$

On a $\mathbf{x}(t+dt) \simeq \mathbf{x}(t) + dt.\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$

(avec dt de l'ordre de e.g. 0.05 s)

Révisions et travaux préliminaires

- Simulation par méthode d'Euler en MATLAB
  - Dans le code MATLAB, on va noter le vecteur d'état x comme un vecteur MATLAB, e.g. pour un modèle char $\mathbf{x} = (x, y, \theta, v)$:

$$\begin{array}{ccc} \text{MATLAB} & \rightarrow & \text{Théorie} \\ \mathbf{x}(1) & \rightarrow & x \\ \mathbf{x}(2) & \rightarrow & y \\ \mathbf{x}(3) & \rightarrow & \theta \\ \mathbf{x}(4) & \rightarrow & v \end{array}$$

  - Les fonctions d'évolution f et d'observation g seront codées comme des fonctions MATLAB : e.g. pour f d'un modèle char

```
function xdot = f(x,u)
xdot = [x(4)*cos(x(3));
    x(4)*sin(x(3));
    u(1);
    u(2)];
end
```

Révisions et travaux préliminaires

- Simulation par méthode d'Euler en MATLAB
  - Une simulation en MATLAB pour un modèle char :

```
x = [0;0;0;10];
u = [0;0];
dt = 0.01;
t = 0;
while (t < 30)
    x = x+f(x,u)*dt;
    clf; draw(x);
    pause(dt);
    t = t+dt;
end
```

Révisions et travaux préliminaires

- Simulation par méthode d'Euler en MATLAB
  - Une simulation en MATLAB pour un modèle char :

Etat initial

```
x = [0;0;0;10];
u = [0;0];
dt = 0.01;
t = 0;
while (t < 30)
    x = x+f(x,u)*dt;
    clf; draw(x);
    pause(dt);
    t = t+dt;
end
```

Intégration de la fonction d'évolution avec Euler pour estimer l'état à t+dt

Appel à une fonction de dessin à définir…

Efface la fenêtre

Attente, pour s'approcher du temps réel

Révisions et travaux préliminaires

- Coordonnées homogènes
  - Pour dessiner un système simulé, on a souvent des **rotations** et des **translations** d'éléments à faire
  - Pour combiner ces 2 types d'opérations facilement, on peut utiliser le formalisme des **coordonnées homogènes**

    e.g. si on veut faire une rotation de theta puis une translation de x,y, il nous faut définir la matrice

$$\mathbf{R} = \begin{pmatrix} \cos\theta & -\sin\theta & x \\ \sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{pmatrix}$$

Révisions et travaux préliminaires

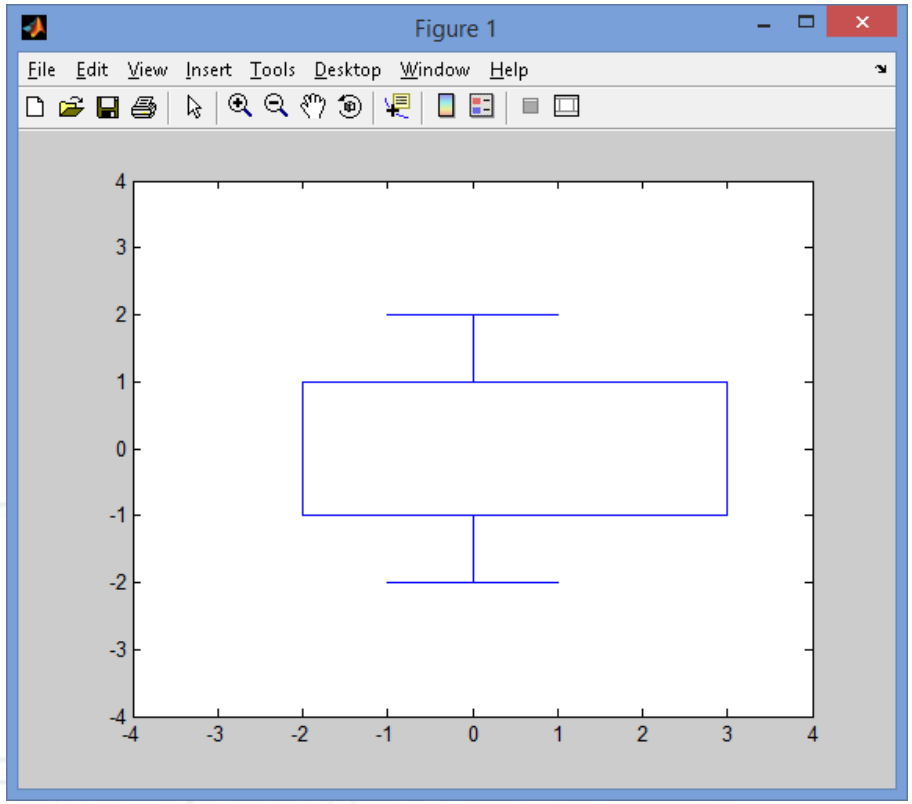
- Coordonnées homogènes
  - Le motif 2D des lignes représentant notre système au repos (**x**=**0**) devra être défini comme une matrice

    M=[coordonnées x…;coordonnées y…;1…]
  - Les 1 de la 3ème colonne sont nécessaires pour que la multiplication par la matrice **R** fonctionne comme attendu

Révisions et travaux préliminaires

- Coordonnées homogènes en MATLAB
  - Dans le code MATLAB, e.g. pour un modèle char pour **x**=[0;0;0]

```
function draw(x)
M = [1 -1  0  0 -2 -2 0 0 -1 1 0 0 3  3  0;
    -2 -2 -2 -1 -1  1 1 2  2 2 2 1 1 -1 -1;
    1  1  1  1  1  1 1 1  1 1 1 1 1  1  1];
R = [ cos(x(3)) -sin(x(3))  x(1);
      sin(x(3))  cos(x(3))  x(2);
      0          0          1];
M = R*M;
plot(M(1,:),M(2,:),'b');
end
```



Révisions et travaux préliminaires

- Coordonnées homogènes en MATLAB
  - Fonctions MATLAB souvent utiles pour le dessin

```
figure; % Create a new figure
clf; % Clear the figure
hold on; % Ensure all the plot() go on the same figure
axis([-5,5,-5,5]); % Set the limits on the axis
axis square; % Make the axis orthonormal
```

Révisions et travaux préliminaires

- Observateur d'état
  - Sur un système réel, il est parfois difficile de mesurer correctement ou directement certaines variables : capteurs imprécis, trop coûteux, pas utilisables des les conditions où on est…

    Exemple : Le GPS marche bien quand on roule sur une route dégagée mais dans un long tunnel il ne marche pas du tout

  - Il faut donc trouver des méthodes théoriques et numériques pour évaluer ces valeurs : on appelle **observateur d'état** ces méthodes

Révisions et travaux préliminaires

- Observateur d'état
  - En automatique 1A, vous avez peut-être vu les **observateurs de Luenberger** pour les **systèmes linéaires invariants**, on peut en faire d'autres selon les spécificités du système étudié :

    Dead-reckoning (navigation à l'estime) : utilisation des équations d'état pour prédire la position à t+dt à partir de celle à t

    Calculs par **relations géométriques** : relations de distance et/ou d'angles à des amers (objets de position plus ou moins connue)

    Filtre de Kalman : détaillé dans la suite…

  - Note : souvent en robotique, on va confondre **estimation d'état**, **observateur d'état** avec **localisation**, **estimation de position** car bien souvent c'est la position qui nous intéresse parmi les variables d'état

Vocabulaire et notions prérequises

Vocabulaire et notions prérequises

- But
  - Pour pouvoir modéliser les erreurs des capteurs, du modèle, etc., nous allons avoir besoin de définir le plus clairement possible certaines de leurs caractéristiques…

Vocabulaire et notions prérequises

- Déterministe vs aléatoire
  - Un système déterministe va toujours produire la même sortie dans les mêmes conditions
  - Ce n’est pas le cas pour un système aléatoire. Cependant, il existe diverses notions pour caractériser ce type de système…

Vocabulaire et notions prérequises

- Variable aléatoire $x \in \mathbb{R}$
  - Fonction de répartition $F : \mathbb{R} \to [0,1], F(z) = P(x \leq z)$
  - Densité / loi de probabilité $p(z) = \frac{dF(z)}{dz}$ soit $p(z)dz = P(z \leq x \leq z + dz)$
  - Espérance / moyenne $m = \bar{x} = E(x) = \int_{-\infty}^{+\infty} zp(z)dz = \int_{-\infty}^{+\infty} zdF(z)$
  - Moment d'ordre k $m_k = E(x^k) = \int_{-\infty}^{+\infty} z^k p(z)dz$
  - Moment centré d'ordre k $E\left((x - \bar{x})^k\right) = \int_{-\infty}^{+\infty} (z - \bar{x})^k p(z)dz$
  - Variance : moment centré d'ordre 2
  - Ecart-type σ : racine carrée de la variance

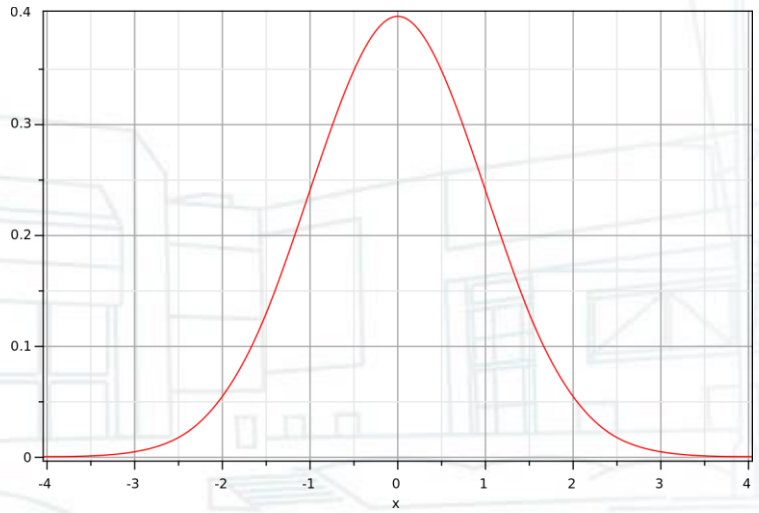
Vocabulaire et notions prérequises

- Variable aléatoire $x \in \mathbb{R}$
  - Variable centrée réduite

    Espérance à 0

    Ecart-type à 1

  - Variable gaussienne / loi normale $p(z) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{(z-\bar{x})^2}{2\sigma^2}}$



Vocabulaire et notions prérequises

- Variable aléatoire vectorielle $\mathbf{x} \in \mathbb{R}^n$
  - Fonction de répartition $\mathbf{F}(\mathbf{z}) = P\left(x_1 \leq z_1 \text{ and } x_2 \leq z_2 \text{ and } \ldots \text{ and } x_n \leq z_n\right)$
  - Matrice de covariance $\mathbf{\Gamma} = E\left((\mathbf{x} - \bar{\mathbf{x}}).(\mathbf{x} - \bar{\mathbf{x}})^{\mathsf{T}}\right)$:

    Matrice dont les éléments sont

    $$\gamma_{ij} = E((x_i - \bar{x}_i)(x_j - \bar{x}_j))$$

    Une matrice de covariance est toujours définie positive (donc carré, symétrique, inversible…)
  - Blancheur : matrice de covariance diagonale, toutes les composantes de la variable vectorielle sont dites indépendantes

Vocabulaire et notions prérequises

- Variable aléatoire vectorielle
  - Variables $\mathbf{x} \in \mathbb{R}^n$ et $\mathbf{y} \in \mathbb{R}^n$ indépendantes :

    Matrice de covariance $\boldsymbol{\Gamma}_{\mathbf{xy}} = E\left((\mathbf{x} - \bar{\mathbf{x}}).(\mathbf{y} - \bar{\mathbf{y}})^{\mathsf{T}}\right) = \mathbf{0}$

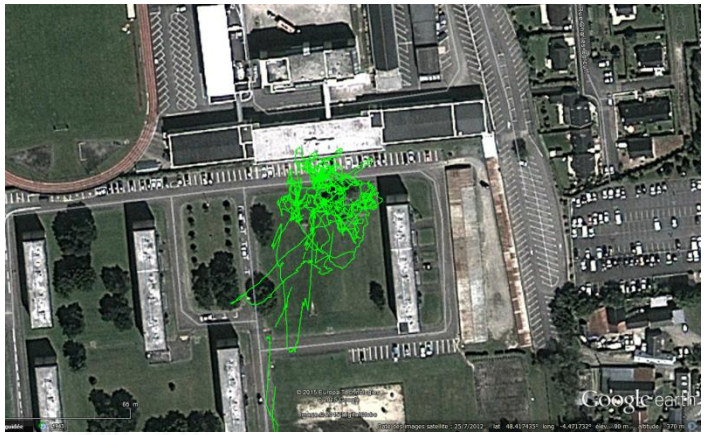
  - Gaussienne, notée $\mathcal{N}(\bar{\mathbf{x}}, \boldsymbol{\Gamma})$

$$\mathbf{p}(\mathbf{z}) = \frac{(2\pi)^{-\frac{n}{2}}}{\sqrt{\det(\boldsymbol{\Gamma})}} e^{\left(-\frac{1}{2}(\mathbf{z}-\bar{\mathbf{x}})^{\mathsf{T}}\boldsymbol{\Gamma}^{-1}(\mathbf{z}-\bar{\mathbf{x}})\right)}$$

Vocabulaire et notions prérequises

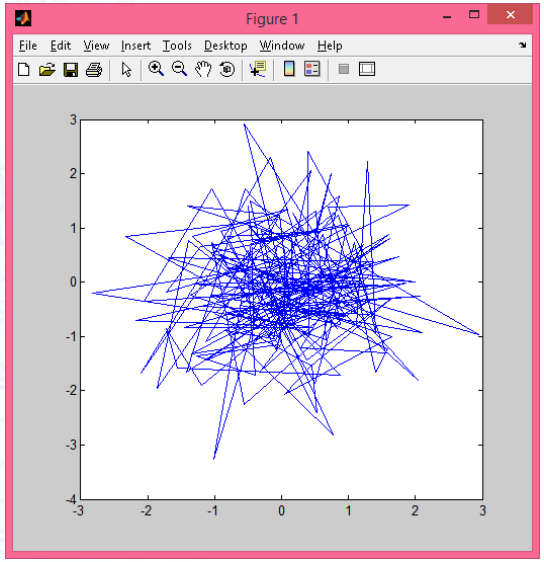
- Signal, MATLAB
  - Un signal aléatoire est une fonction du temps qui associe à chaque instant une valeur d'une variable ou d'un vecteur aléatoire

    Exemple : position donnée par un GPS



  - En MATLAB, un vecteur aléatoire peut être représenté par un nuage de points associé à des réalisations (e.g. chaque réalisation ayant été produite à un t donné)

    Exemple : nuage de points (connectés) 2D gaussien avec $\boldsymbol{\Gamma} = \mathbf{I}$ et $\bar{\mathbf{x}} = \mathbf{0}$




```
x1=randn(200,1);x2=randn(200,1);plot(x1,x2)
```

Vocabulaire et notions prérequises

- Vecteurs aléatoires en MATLAB
  - Retrouver $\bar{\mathbf{x}}$ et Γ à partir des vecteurs aléatoires

```
xbar=mean([x1 x2])'
G=cov([x1 x2])
```

  - Ellipsoïde de confiance

    Pour un vecteur aléatoire 2D, on peut dessiner la zone de probabilité d'appartenance η avec le code suivant

```
function ellipsoid(xbar, G, eta)
s=0:0.01:2*pi;
w=xbar*ones(size(s))+sqrtm(-2*log(1-eta)*G)*[cos(s);sin(s)];
plot(w(1,:),w(2,:));
end
```

Vocabulaire et notions prérequises

- Génération de vecteurs aléatoires gaussiens

Théorème. Si $\mathbf{x}$, $\boldsymbol{\alpha}$ et $\mathbf{y}$ sont 3 vecteurs aléatoires liés par la relation $\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\alpha} + \mathbf{b}$ où $\mathbf{A}$ et $\mathbf{b}$ sont déterministes, que $\mathbf{x}$, $\boldsymbol{\alpha}$ sont indépendants et que $\boldsymbol{\alpha}$ est centré, on a

$$\begin{aligned} \bar{\mathbf{y}} &= \mathbf{A}\bar{\mathbf{x}} + \mathbf{b} \\ \boldsymbol{\Gamma}_{\mathbf{y}} &= \mathbf{A}.\boldsymbol{\Gamma}_{\mathbf{x}}.\mathbf{A}^{\mathrm{T}} + \boldsymbol{\Gamma}_{\boldsymbol{\alpha}} \end{aligned}$$

Exemple. Si $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, le vecteur aléatoire $\mathbf{y} = \boldsymbol{\Gamma}_{\mathbf{y}}^{1/2}\mathbf{x} + \bar{\mathbf{y}}$ aura une espérance de $\bar{\mathbf{y}}$ et une matrice de covariance égale à $\boldsymbol{\Gamma}_{\mathbf{y}}$ (vu que $\mathbf{A} = \boldsymbol{\Gamma}_{\mathbf{y}}^{1/2}$ et $\mathbf{b} = \bar{\mathbf{y}}$ en remplaçant dans le théorème).

- => On utilise cette propriété pour générer en MATLAB des vecteurs gaussiens non centrés réduits ($\bar{\mathbf{y}} \neq \mathbf{0}$ et $\boldsymbol{\Gamma}_{\mathbf{y}} \neq \mathbf{I}$)

  Exemple

```
n=1000;Gy=[3,1;1,3];ybar=[2;3];x=randn(size(ybar,1),n);
y=ybar*ones(1,n)+sqrtm(Gy)*x;
plot(y(1,:),y(2,:),'.');
```

- Une fonction MATLAB est aussi disponible : **mvnrnd** (nécessite **Statistics and machine learning** toolbox)

Formalisation du filtre de Kalman et utilisation

Formalisation du filtre de Kalman et utilisation

- Description rapide
  - Le **filtre de Kalman** (=Linear Quadratic Estimation) est un algorithme qui **estime l'état** d'un système **à partir de mesures** incomplètes ou bruitées **et d'un modèle d'état**
  - L'état peut être calculé pour l'instant présent (filtrage/correction/mise à jour/innovation), un instant passé (lissage), ou sur un horizon futur (prédiction)
  - Une **estimation de l'erreur** est aussi fournie
  - En général seule la **prédiction** et **mise à jour** sont faites quand on l'utilise comme observateur d'état sur un système réel

Formalisation du filtre de Kalman et utilisation

- Description rapide
  - Pour que le filtre de Kalman fonctionne bien, il faut que le système respecte un certain nombre d'hypothèses et contraintes, notamment sur la forme de ses équations d'état et les erreurs des mesures et du modèle

Formalisation du filtre de Kalman et utilisation

- Forme des équations d'état considérées

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{A}_k \mathbf{x}_k + \mathbf{u}_k + \boldsymbol{\alpha}_k \\ \mathbf{y}_k = \mathbf{C}_k \mathbf{x}_k + \boldsymbol{\beta}_k \end{cases}$$

- $\boldsymbol{\alpha}_k$ et $\boldsymbol{\beta}_k$ aléatoires gaussiens centrés indépendants entre eux et blancs dans le temps ($k_1 \neq k_2 \Rightarrow \boldsymbol{\Gamma}_{\boldsymbol{\alpha}_{k_1}\boldsymbol{\alpha}_{k_2}} = E\left((\boldsymbol{\alpha}_{k_1} - \bar{\boldsymbol{\alpha}}_{k_1}).(\boldsymbol{\alpha}_{k_2} - \bar{\boldsymbol{\alpha}}_{k_2})^{\mathsf{T}}\right) = \mathbf{0}$)
- $\boldsymbol{\alpha}_k$ bruit d'état : si l'équation d'évolution ne représente pas très bien la réalité du système
- $\boldsymbol{\beta}_k$ bruit de mesure : si on sait que les mesures sont imprécises

Formalisation du filtre de Kalman et utilisation

- Idée de la démonstration
  - On cherche la meilleure façon d'estimer xk à partir de yk (mesures actuelles) et xk-1 (estimation de l'état précédent)
  - Une idée est de prendre une sorte de moyenne pondérée entre les mesures actuelles et la prédiction donnée par la méthode d'Euler avec les équations d'état
  - En formalisant, la forme de notre estimateur devrait être

$$\hat{\mathbf{x}}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{C}_k)(\mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1} + \mathbf{u}_{k-1}) + \mathbf{K}_k\mathbf{y}_k$$

avec K un poids à choisir pour la moyenne : il sera choisi pour minimiser les erreurs au sens des moindres carrés

  - Plus d'infos sur la démo, voir Part 7.2, 7.3.1, 7.4 sur :
    https://www.ensta-bretagne.fr/jaulin/poly_kalman.pdf

Formalisation du filtre de Kalman et utilisation

- Notations $\hat{\mathbf{x}}_{k|k-1}$, $\hat{\mathbf{x}}_{k|k}$ …
  - $\hat{\mathbf{x}}_{k|k-1}$: vecteur aléatoire représentant l'estimation de l'état après la prise en compte de $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{k-1}$, $\mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_{k-1}$, $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{k-1}$
  - $\hat{\mathbf{x}}_{k|k}$: prise en compte de $\mathbf{y}_k$, $\mathbf{C}_k$ en plus
  - $\mathbf{A}_k$, $\mathbf{C}_k$: peuvent dépendre de k, i.e. du temps, mais pas de l'état (équations d'état linéaires w.r.t. variables d'état)

Formalisation du filtre de Kalman et utilisation

- Equations du filtre de Kalman

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{A}_k \hat{\mathbf{x}}_{k|k} + \mathbf{u}_k \quad \text{(estimée prédite)}$$

$$\mathbf{\Gamma}_{k+1|k} = \mathbf{A}_k . \mathbf{\Gamma}_{k|k} . \mathbf{A}_k^{\mathrm{T}} + \mathbf{\Gamma}_{\alpha_k} \quad \text{(covariance prédite)}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k . \widetilde{\mathbf{y}}_k \quad \text{(estimée corrigée)}$$

$$\mathbf{\Gamma}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{C}_k)\, \mathbf{\Gamma}_{k|k-1} \quad \text{(covariance corrigée)}$$

$$\widetilde{\mathbf{y}}_k = \mathbf{y}_k - \mathbf{C}_k \hat{\mathbf{x}}_{k|k-1} \quad \text{(innovation)}$$

$$\mathbf{S}_k = \mathbf{C}_k \mathbf{\Gamma}_{k|k-1} \mathbf{C}_k^{\mathrm{T}} + \mathbf{\Gamma}_{\beta_k} \quad \text{(covariance de l'innovation)}$$

$$\mathbf{K}_k = \mathbf{\Gamma}_{k|k-1} \mathbf{C}_k^{\mathrm{T}} \mathbf{S}_k^{-1} \quad \text{(gain de Kalman)}$$

(L'ordre ici n'est pas forcément celui dans lequel on fait les calculs)

Variable intermédiaire
Mise à jour
Prédiction

Formalisation du filtre de Kalman et utilisation

- Détails des différentes parties
  - Mise à jour à l'instant k :

    On vient de recevoir des mesures dans $\mathbf{y}_k$

    On a $\hat{\mathbf{x}}_{k|k-1}$ (prédiction faite à l'instant k-1)

    => On va chercher à calculer $\hat{\mathbf{x}}_{k|k}$ (qui prendra donc en compte $\mathbf{y}_k$ et $\hat{\mathbf{x}}_{k|k-1}$)

    => En parallèle, les matrices de covariance des différentes variables sont aussi calculées (pour représenter les erreurs)

Formalisation du filtre de Kalman et utilisation

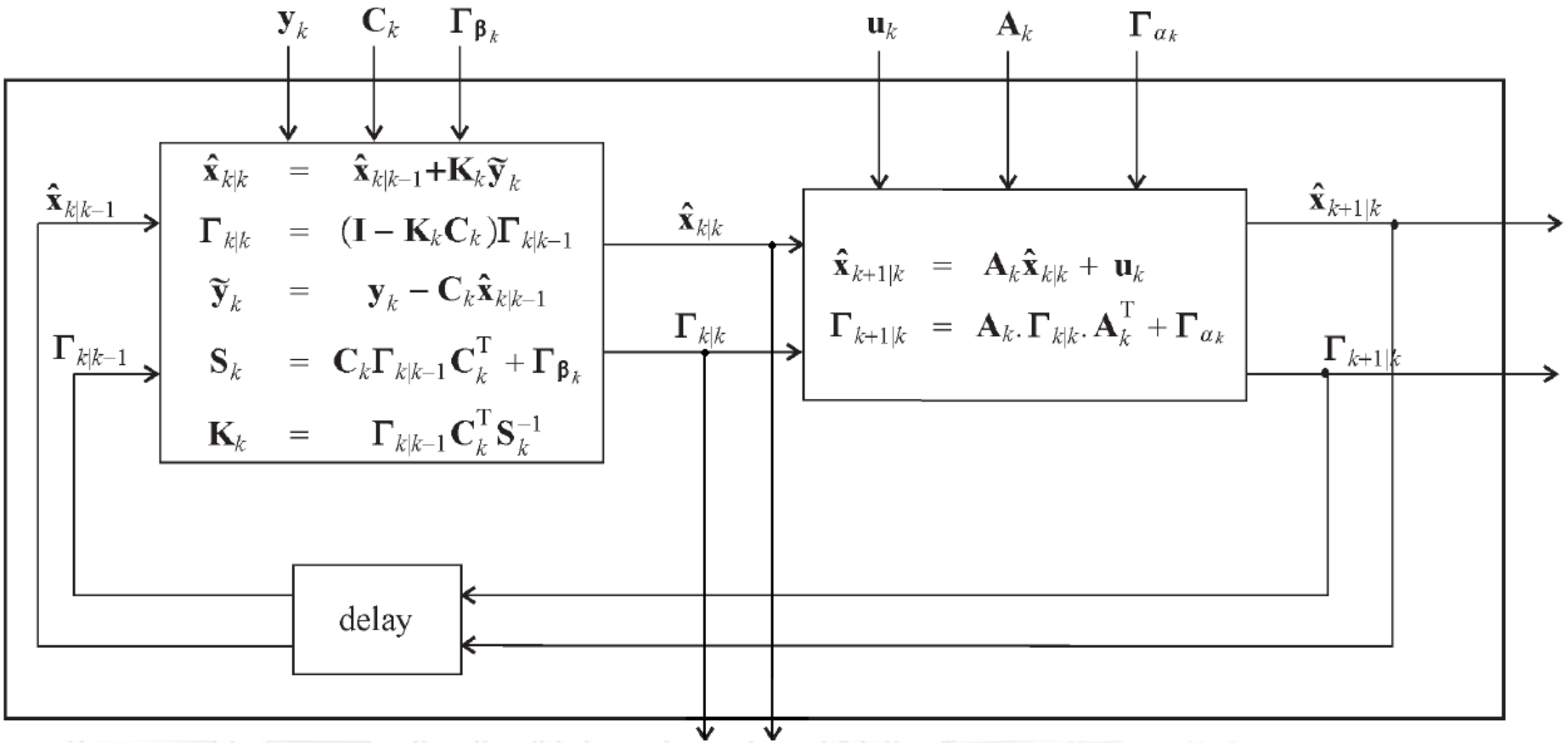
- Détails des différentes parties
  - Prédiction de l'instant k+1 à l'instant k :

    On vient de calculer $\hat{\mathbf{x}}_{k|k}$

    => On va chercher à calculer $\hat{\mathbf{x}}_{k+1|k}$ grâce à l'équation d'évolution (prendra donc en compte $\mathbf{y}_k$ via $\hat{\mathbf{x}}_{k|k}$, mais pas $\mathbf{y}_{k+1}$ car on ne l'a pas encore)

    => En parallèle, les matrices de covariance des différentes variables sont aussi calculées (pour représenter les erreurs)

Formalisation du filtre de Kalman et utilisation



Formalisation du filtre de Kalman et utilisation

- Fonction MATLAB

```
function [x1,G1,xup,Gup]=kalman(x0,G0,u,y,Galpha,Gbeta,A,C)
      if (isempty(y)), % When no output (predictor),
          n=length(x0);
          y=eye(0,1); Gbeta=eye(0,0); C=eye(0,n);
      end;
      S=C*G0*C'+Gbeta;
      K=G0*C'*inv(S);
      ytilde=y-C*x0;
      xup=x0+K*ytilde; % up = update
      Gup=G0-K*C*G0;
      x1=A*xup + u;
      G1=A*Gup*A'+Galpha;
end
```

MATLAB	$\rightarrow$	Théorie
x0	$\rightarrow$	$\hat{\mathbf{x}}_{k\|k-1}$
G0	$\rightarrow$	$\mathbf{\Gamma}_{k\|k-1}$
x1	$\rightarrow$	$\hat{\mathbf{x}}_{k+1\|k}$
G1	$\rightarrow$	$\mathbf{\Gamma}_{k+1\|k}$
xup	$\rightarrow$	$\hat{\mathbf{x}}_{k\|k}$
Gup	$\rightarrow$	$\mathbf{\Gamma}_{k\|k}$

Valeurs à utiliser à l'instant k pour e.g. le dessin…

Formalisation du filtre de Kalman et utilisation

- Avantages
  - Rectification des erreurs, non seulement des capteurs, mais aussi du modèle
  - Capacité de prédiction de l'état
  - Capacité à déterminer l'erreur moyenne de son estimation :

    Vecteur contenant l'état estimé $\hat{\mathbf{x}}_{k|k}$

    Matrice de covariance de l'erreur $\mathbf{\Gamma}_{k|k}$
  - Dans le cas de **bruits gaussiens** et **équations d'état linéaires**, il est **optimal**

Formalisation du filtre de Kalman et utilisation

- Inconvénients / Alternatives
  - Cas où les équations d'état sont **non-linéaires** : filtre de Kalman étendu (Extended Kalman Filter : **EKF**) => on va devoir faire des linéarisations

    => Matrices jacobiennes à calculer => plus de calculs

    => La covariance de l'erreur (la précision des estimations) ne converge pas obligatoirement (comme c'était le cas avec une modélisation linéaire)

    => Mais en fait en pratique on n'utilise l'EKF qu'en dernier recours, on essaye plutôt de « s'arranger » pour que les équations d'état soient linéaires…
  - Cas où les bruits sont **non gaussiens** ou équations d'état difficiles à trouver : **filtres particulaires** (Monte-Carlo…), filtres ensemblistes (calcul par intervalles…)

Formalisation du filtre de Kalman et utilisation

- Démarche pour adapter les équations d'un problème pour pouvoir utiliser le filtre de Kalman

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{A}_k\mathbf{x}_k + \mathbf{u}_k + \boldsymbol{\alpha}_k \\ \mathbf{y}_k = \mathbf{C}_k\mathbf{x}_k + \boldsymbol{\beta}_k \end{cases}$$

  - **Faire le point sur les variables** qui peuvent être **connues**, celles qu'on souhaite **estimer**, et les **autres** qu'on ne connait pas
  - **Définir un nouveau vecteur d'état** $\mathbf{z}$ ne contenant que des variables à estimer ou inconnues
  - **Réécrire les équations d'état avec ce nouveau vecteur d'état**. Si ces équations ne peuvent pas être écrites sous la forme $\dot{\mathbf{z}} = A\mathbf{z} + \mathbf{u}_z$ avec la matrice A et le vecteur $\mathbf{u}_z$ ne dépendant pas de $\mathbf{z}$, identifier les variables dans $\mathbf{z}$ qui posent problème et rajouter un capteur pour les mesurer et reprendre à 0 la démarche avec ces nouvelles variables connues. Si ce n'est pas possible, revoir les équations d'état initiales, trouver une méthode fiable de linéarisation (voir EKF) ou abandonner le filtre de Kalman…
  - **Discrétiser** et en déduire $\mathbf{A}_k$ et $\mathbf{u}_k$:

$$\begin{aligned} \mathbf{z}_{k+1} &= \mathbf{z}_k + dt \cdot (A\mathbf{z}_k + \mathbf{u}_z) \\ &= \underbrace{(I + dt \cdot A)}_{=\mathbf{A}_k}\ \mathbf{z}_k + \underbrace{(dt \cdot \mathbf{u}_z)}_{=\mathbf{u}_k} \end{aligned}$$

=> Correspond à la partie **prédiction** du filtre de Kalman

Formalisation du filtre de Kalman et utilisation

- Démarche pour adapter les équations d'un problème pour pouvoir utiliser le filtre de Kalman

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{A}_k\mathbf{x}_k + \mathbf{u}_k + \boldsymbol{\alpha}_k \\ \mathbf{y}_k = \mathbf{C}_k\mathbf{x}_k + \boldsymbol{\beta}_k \end{cases}$$

  - De plus, si certaines des variables dans $\mathbf{z}$ (ou des combinaisons linéaires de ces variables) peuvent être mesurées, $\mathbf{y}_k$ devra regrouper ces mesures et $\mathbf{C}_k$ sera la matrice qui permettrait de les retourner si on la multipliait par $\mathbf{z}_k$

=> Correspond à la partie **mise à jour** du filtre de Kalman

Formalisation du filtre de Kalman et utilisation

- Démarche pour adapter les équations d'un problème pour pouvoir utiliser le filtre de Kalman
  - Notes

    Variables **connues** : entrées, variables mesurées précisément par des capteurs directement ou via des relations simples non différentielles, etc.

    Variables **à estimer** : c'est celles qu'on souhaite estimer au mieux, il faut pouvoir exprimer leur dérivée, elles peuvent être mesurées ou non par des capteurs (généralement peu précis, trop rarement fonctionnels, etc.). Si aucune n'est mesurée, seule la partie prédiction du filtre de Kalman pourra être utilisée.

    Variables **inconnues** : celles qui restent...

Formalisation du filtre de Kalman et utilisation

- Démarche pour adapter les équations d'un problème pour pouvoir utiliser le filtre de Kalman
  - Notes

    Matrice de covariance Γ_{α_k} pour spécifier le bruit d'état :

    A régler en fonction des erreurs de prédiction observées, peut dépendre du dt de discrétisation…

    Matrice de covariance Γ_{β_k} pour spécifier le bruit de mesure :

    Parfois la documentation d'un capteur indique son écart-type σ, dans ce cas la variance en est le carré, parfois c'est une erreur qui est évoquée, dans ce cas on peut supposer qu'elle correspond à 3σ (correspond à plus de 99% des valeurs si l'erreur suit une gaussienne) et on en déduit la variance

    On met la variance dans la diagonale de la matrice de covariance à la ligne correspondant au capteur considéré

Formalisation du filtre de Kalman et utilisation

- Démarche pour adapter les équations d'un problème pour pouvoir utiliser le filtre de Kalman
  - Exemple de la voiture

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\delta} \end{pmatrix} = \begin{pmatrix} v\cos\delta\cos\theta \\ v\cos\delta\sin\theta \\ \frac{v\sin\delta}{L} \\ u_1 \\ u_2 \end{pmatrix}$$

u1 et u2 sont connues car ce sont des entrées

x,y est la position à estimer

theta,v,delta sont inconnues à ce stade mais on voit des cos et des sin de theta et delta donc on ne va pas pouvoir les mettre dans le vecteur d'état z sinon on ne va pas pouvoir écrire $\dot{\mathbf{z}} = A\mathbf{z} + \mathbf{u}_z$ avec A ne dépendant pas de z…

=>Il faut donc rajouter un capteur pour connaitre theta et delta (c'est possible avec une boussole et un capteur d'angle)

=>On définit le vecteur d'état intermédiaire $\mathbf{z} = (x, y, v)$ donc on a

$$\dot{\mathbf{z}} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v\cos\delta\cos\theta \\ v\cos\delta\sin\theta \\ u_1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & \cos\delta\cos\theta \\ 0 & 0 & \cos\delta\sin\theta \\ 0 & 0 & 0 \end{pmatrix}\mathbf{z} + \begin{pmatrix} 0 \\ 0 \\ u_1 \end{pmatrix}$$

Ensuite, on discrétise avec Euler :

$$\mathbf{z}_{k+1} = \mathbf{z}_k + dt \cdot \left[\begin{pmatrix} 0 & 0 & \cos\delta_k\cos\theta_k \\ 0 & 0 & \cos\delta_k\sin\theta_k \\ 0 & 0 & 0 \end{pmatrix}\mathbf{z}_k + \begin{pmatrix} 0 \\ 0 \\ u_1(k) \end{pmatrix} \right] \Rightarrow \mathbf{z}_{k+1} = \underbrace{\begin{pmatrix} 1 & 0 & dt\cos\delta_k\cos\theta_k \\ 0 & 1 & dt\cos\delta_k\sin\theta_k \\ 0 & 0 & 1 \end{pmatrix}}_{=\mathbf{A}_k}\mathbf{z}_k + \underbrace{\begin{pmatrix} 0 \\ 0 \\ dt \cdot u_1(k) \end{pmatrix}}_{=\mathbf{u}_k}$$

Formalisation du filtre de Kalman et utilisation

- Démarche pour adapter les équations d'un problème pour pouvoir utiliser le filtre de Kalman
  - De plus, si la voiture a un GPS (mesure x,y) et des odomètres (mesurent v), on a

$$\mathbf{y}_k = \begin{pmatrix} x \\ y \\ v \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{=\mathbf{C}_k} \mathbf{z}_k$$

  - On pourra donc utiliser la partie mise à jour/correction du filtre de Kalman
  - Note : theta et delta sont aussi mesurés mais comme ils sont considérés comme parfaitement connus et ne sont pas dans zk (qui contient les variables que le Kalman va chercher à estimer au mieux), ils ne sont pas à mettre dans yk. Il se peut aussi que toutes les variables de zk ne soient pas mesurées et dans ce cas seule la partie prédiction du Kalman permet de les estimer

Formalisation du filtre de Kalman et utilisation

- Démarche pour adapter les équations d'un problème pour pouvoir utiliser le filtre de Kalman

```
zk= ? % Initial estimation
Gzk= ? % Corresponding error as a covariance matrix
% ...
while ()
    % ...
    uk= ? % Kalman filter known inputs
    Ak= ? % Kalman filter evolution matrix
    Galphak = ? % Model error as a covariance matrix
    yk= ? % Direct measurements of parts of zk (set to [] if not available)
    Ck= ? % Kalman filter observation matrix
    Gbetak= ? % Measurements error as a covariance matrix
    % Computations with the Kalman filter.
    [zk,Gzk,zkup,Gzkup]=kalman(zk,Gzk,uk,yk,Galphak,Gbetak,Ak,Ck);
    %...
    % Ellipse showing the estimated position and error if [zk(1);zk(2)]
    % corresponds to the position
    draw_ellipse([zkup(1);zkup(2)],[Gzkup(1,1) Gzkup(1,2);Gzkup(2,1) Gzkup(2,2)],0.9)
    %...
end
```

Extended Kalman Filter

Extended Kalman Filter

- Extended Kalman Filter (EKF)
  - Lorsque les équations d'état du système étudié ne sont pas linéaires et qu'on doit utiliser une méthode de linéarisation (e.g. autour d'un point de fonctionnement) pour pouvoir utiliser le filtre de Kalman, on parle souvent d'Extended Kalman Filter
  - Dans ce cas, une petite adaptation de la fonction kalman vue précédemment peut être faite…

Extended Kalman Filter

- Extended Kalman Filter (EKF)
  - Equations d'état du système :

$$\begin{cases} \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ \mathbf{y}(t) &= \mathbf{g}(\mathbf{x}(t)) \end{cases}$$

  - Linéarisation autour d'un point de fonctionnement $(\mathbf{x}_p, \mathbf{u}_p)$:

$$\begin{cases} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}_p, \mathbf{u}_p) + \mathbf{A}(\mathbf{x} - \mathbf{x}_p) + \mathbf{B}(\mathbf{u} - \mathbf{u}_p) \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}_p) + \mathbf{C}(\mathbf{x} - \mathbf{x}_p) \end{cases}$$

$$\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_p, \mathbf{u}_p) \qquad \mathbf{B} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}(\mathbf{x}_p, \mathbf{u}_p) \qquad \mathbf{C} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\mathbf{x}_p)$$

  - Euler : $\mathbf{x}_{k+1} = \mathbf{x}_k + dt.(\mathbf{f}(\mathbf{x}_p, \mathbf{u}_p) + \mathbf{A}(\mathbf{x}_k - \mathbf{x}_p) + \mathbf{B}(\mathbf{u} - \mathbf{u}_p))$
  - On peut retomber sur les équations pour le filtre de Kalman :

$$\begin{cases} \mathbf{x}_{k+1} &= \mathbf{A}_k \mathbf{x}_k + \mathbf{u}_k \\ \mathbf{z}_k &= \mathbf{C}_k \mathbf{x}_k \end{cases} \quad \text{avec}$$

$$\mathbf{A}_k = \mathbf{I} + dt.\mathbf{A}$$
$$\mathbf{u}_k = dt.\left(\mathbf{f}(\mathbf{x}_p, \mathbf{u}_p) - \mathbf{A}\mathbf{x}_p + \mathbf{B}(\mathbf{u} - \mathbf{u}_p)\right)$$
$$\mathbf{C}_k = \mathbf{C} \text{ si } \mathbf{z}_k = \mathbf{y}_k - \mathbf{g}(\mathbf{x}_p) + \mathbf{C}\mathbf{x}_p$$

Extended Kalman Filter

- Extended Kalman Filter (EKF)
  - Fonction MATLAB kalman_ekf adaptée au cas non linéaire :

```
function [x1,G1,xup,Gup]=kalman_ekf(x0,G0,u,y,Galpha,Gbeta,A,C,dt)
      if (isempty(y)), % When not output (predictor),
          n=length(x0);
          y=eye(0,1); Gbeta=eye(0,0); C=eye(0,n);
      end;
      S=C*G0*C'+Gbeta;
      K=G0*C'/S;
      %ytilde=y-C*x0; % Simple Kalman filter
      ytilde=y-g(x0); % EKF
      xup=x0+K*ytilde; % up = update
      Gup=G0-K*C*G0;
      %x1=A*xup + u; % Simple Kalman filter
      x1=xup+dt*f(xup,u); % EKF
      G1=A*Gup*A'+Galpha;
end
```

Extended Kalman Filter

- Extended Kalman Filter (EKF)
  - Différences des paramètres des fonctions MATLAB kalman_ekf et kalman :

    Paramètre dt en plus pour kalman_ekf

    Paramètre y : si on a dû linéariser g et qu'on souhaite utiliser la fonction kalman, il faut passer $\mathbf{z}_k = \mathbf{y}_k - \mathbf{g}(\mathbf{x}_p) + \mathbf{C}\mathbf{x}_p$ au paramètre y, par contre si on utilise kalman_ekf on passe directement y

Lisseur de Kalman

- Lisseur de Kalman
  - Voir les 2 dernières pages du poly : https://www.ensta-bretagne.fr/jaulin/poly_kalman.pdf