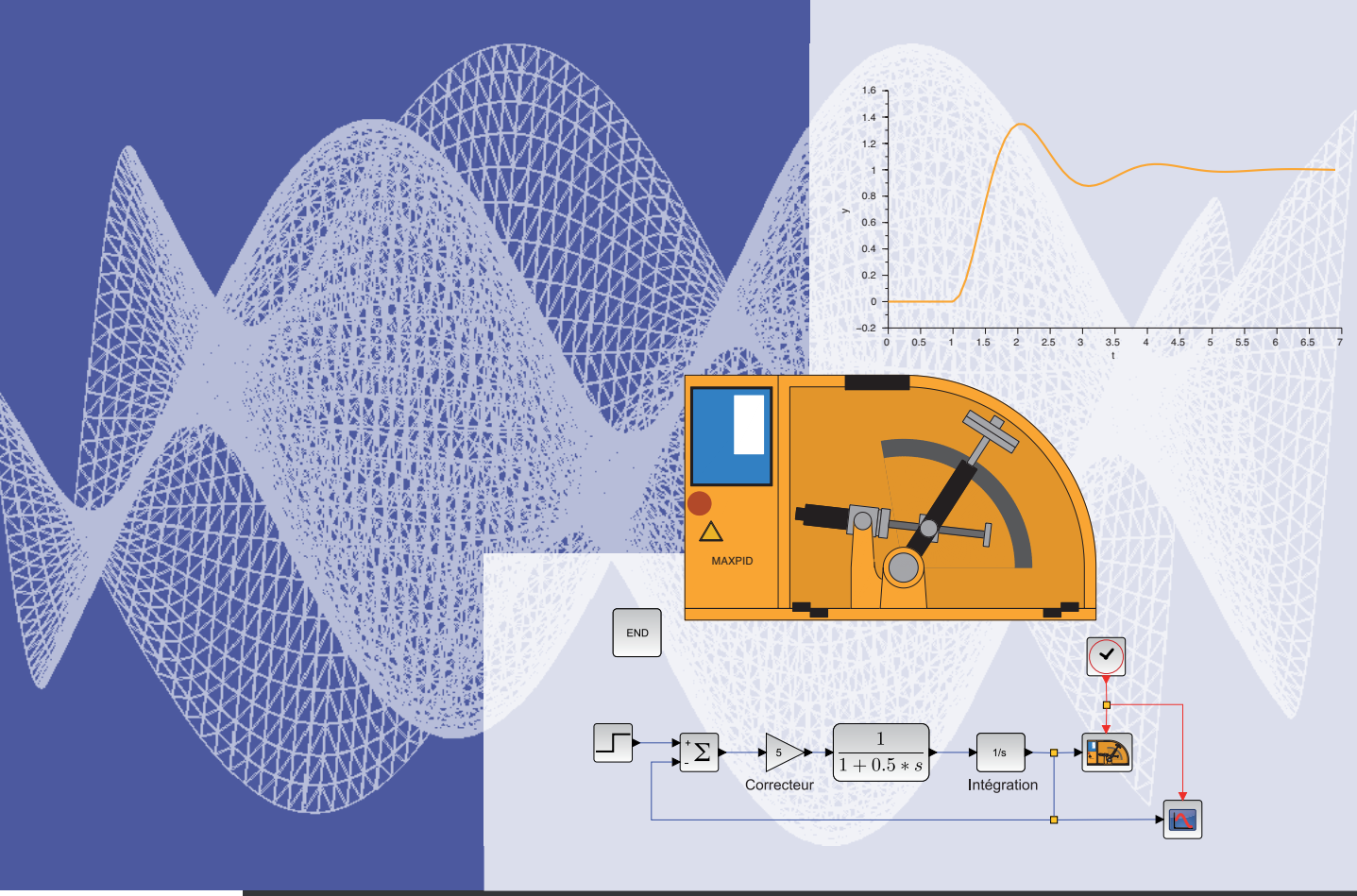


Scilab

Scilab / Xcos
pour l'enseignement des sciences de l'ingénieur

Réalisé avec le soutien d'Inria, ce livret a été co-écrit par Scilab Enterprises et Messieurs Alain Caignot, Lycée Stanislas (Paris), Vincent Crespel, Lycée Saint Louis (Paris), Marc Derumaux, Lycée Saint Louis (Paris), Cédric Lusseau, Lycée Hoche (Versailles), Gilles Moissard, Lycée Janson de Sailly (Paris), Pascal Serrier, Lycée Benjamin Franklin (Orléans) et David Violeau, Lycée Janson de Sailly (Paris).

Scilab-Xcos
pour l'enseignement des sciences de l'ingénieur

Introduction

1- Se familiariser à Xcos

2 - Modélisation multi physique acausale (Module SIMM)

3 - Modélisation et analyse de systèmes à temps continu (Module CPGE)

4 - Acquisition et pilotage de moteur (Module Arduino)

INTRODUCTION

À PROPOS DE CE LIVRET

L'objectif de ce livret est de vous guider pas à pas dans la découverte des différentes fonctionnalités de base de l'outil Xcos inclus dans Scilab dans le cadre d'une utilisation en classe de sciences de l'ingénieur.

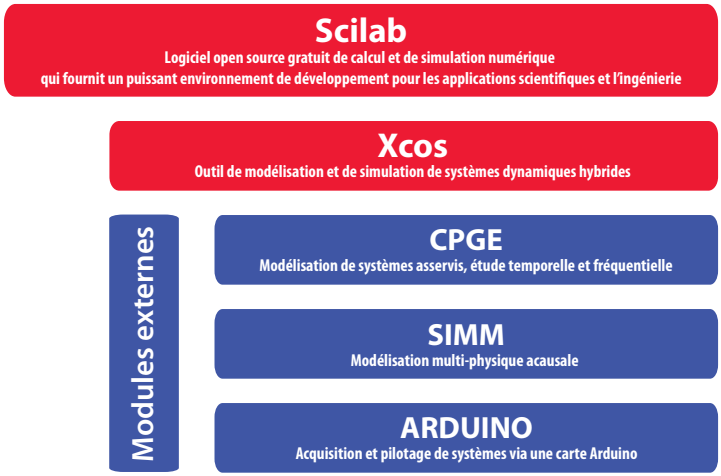
Cette présentation propose plusieurs exemples d'applications qui permettent de découvrir les fonctionnalités utiles à l'enseignement et à l'apprentissage des sciences de l'ingénieur en section S-SI ou STIDD, au lycée et dans l'enseignement supérieur.

Les exemples, diagrammes et illustrations sont réalisés avec Scilab 5.4.1 enrichi des modules CPGE, SIMM et Arduino. Vous pouvez donc reproduire tous les exemples présentés à partir de la version 5.4.1 de Scilab (les diagrammes sont mis à disposition depuis la barre de menus en cliquant sur **? / Démonstrations / nom du module correspondant / Exemples Livret**). Il est indispensable de disposer d'un compilateur C pour utiliser le module de modélisation multi-physique SIMM (se référer au sous-chapitre « Installation d'un compilateur C »).

INSTALLATION DE SCILAB ET CONFIGURATIONS UTILES EN SCIENCES DE L'INGÉNIEUR

Scilab est un logiciel open source de calcul numérique que chacun peut télécharger gratuitement. Disponible sous Windows, Linux et Mac OS X, Scilab est téléchargeable à l'adresse suivante : http://www.scilab.org

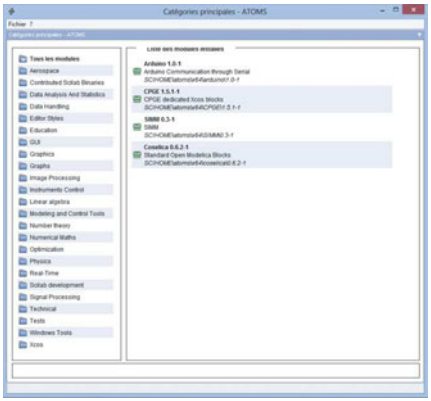
Une fois Scilab téléchargé et installé, il faut ajouter les modules externes complémentaires, listés ci-dessous :



À noter
Lors de l'installation, Scilab va télécharger des fichiers qui seront stockés avec l'exécutable d'installation. S'ils sont détectés lors d'une nouvelle installation, ils ne seront pas téléchargés de nouveau, ce qui est utile pour une installation dans une salle sans accès Internet.

Pour cela, il suffit :

- D'être connecté à Internet
- De lancer Scilab
- De cliquer dans la barre de menus sur **Applications / Gestionnaire de modules – ATOMS** puis d'aller chercher les différents modules dans les catégories indiquées ci-dessous :
  - CPGE (catégorie Éducation),
  - SIMM (catégorie Éducation),
  - Arduino (catégorie Instruments Control).



Pour chaque module, cliquez sur le bouton **Installer**, quittez Scilab et relancez-le. Les modules sont alors installés et se chargeront automatiquement à chaque démarrage de Scilab.

Pour supprimer un module ou le mettre à jour, il suffit de retourner dans le **Gestionnaire de modules – ATOMS** et de cliquer cette fois sur le bouton correspondant **Supprimer** ou **Mettre à jour**.

Si vous ne disposez pas d'une connexion Internet, connectez-vous depuis un autre poste relié à Internet et téléchargez les différents modules (fichiers .bin.zip) à partir du site : http://atoms.scilab.org

Tapez dans la console :

```
--> atomsSetConfig("offLine","True")
--> atomsInstall("chemin_de_telechargement\nom_du_fichier_zip_telecharge")
```

Par exemple sous Windows, si le module CPGE, sous la forme d'un fichier zip, a été téléchargé dans E:\Telechargement, on tape la commande :

```
--> atomsInstall("E:\Telechargement\nom_du_fichier.bin.zip")
```

Installation d'un compilateur C

- **Sous Windows**

  Installez depuis le **Gestionnaire de modules – ATOMS**, le module MinGW (catégorie Windows Tools). Suivez bien la procédure décrite dans la fenêtre d'installation du module.

À noter
Dans le cas d'un environnement Unix (Linux ou Mac OS), remplacer les anti-slashs (\) par des slashs (/) conformément aux habitudes d'écriture des chemins d'accès aux fichiers.

- **Sous Linux**
  Le compilateur GCC étant disponible dans ce système d'exploitation, il suffit de vérifier (via Synaptic, Yum ou tout autre système de gestion de paquets) qu'il est présent et à jour.
- **Sous Mac**
  Téléchargez XCode via l'App Store (Mac OS ≥ 10.7) ou via les CD fournis avec l'ordinateur (Mac OS 10.5 et 10.6). Pour les versions antérieures, voir le site d'Apple. Validez la possibilité d'utiliser le compilateur hors de l'environnement Xcode. Pour cela, après avoir lancé Xcode, allez dans « Préférences », puis « Downloads » et, dans l'onglet « Components », cochez la case « Check for and install updates automatically » et installez l'extension « Command Line Tools ».

Il est bien entendu que si un compilateur C est déjà installé sur votre machine, il n'est pas nécessaire d'en installer un nouveau. Pour vérifier que Scilab a bien détecté votre compilateur, utilisez la commande qui retourne %T si un compilateur est installé :

```
--> haveacompiler()
```

Liste de diffusion

Pour faciliter l'échange entre les utilisateurs de Scilab du monde de l'éducation, une liste de diffusion leur est dédiée. Le principe est simple. Les personnes inscrites peuvent communiquer les unes avec les autres par courrier électronique (questions, réponses, partage de documents, retour d'expériences...).
Pour s'inscrire, il suffit de compléter un formulaire disponible en ligne à l'adresse suivante :
http://lists.scilab.org/mailman/listinfo/enseignement

Vous recevrez une confirmation de votre inscription. Il vous suffira alors d'envoyer un message à l'adresse enseignement@lists.scilab.org pour que celui-ci soit redistribué automatiquement à tous les inscrits de la liste.

Ressources complémentaires

Le site Internet de Scilab dispose d'une rubrique consacrée à l'utilisation de Scilab pour l'enseignement (http://www.scilab.org/fr/community/education), avec des liens et des documents utiles, dont le présent livret au format PDF, un livret destiné à l'enseignement des mathématiques, des exercices et des corrigés d'épreuves pratiques, le tout pouvant être téléchargé et imprimé librement.

Le site des professeurs ayant largement contribué à l'écriture de ce présent livret est également une source riche d'informations et d'exemples d'utilisation : http://www.demosciences.fr/

1 - Se familiariser à Xcos

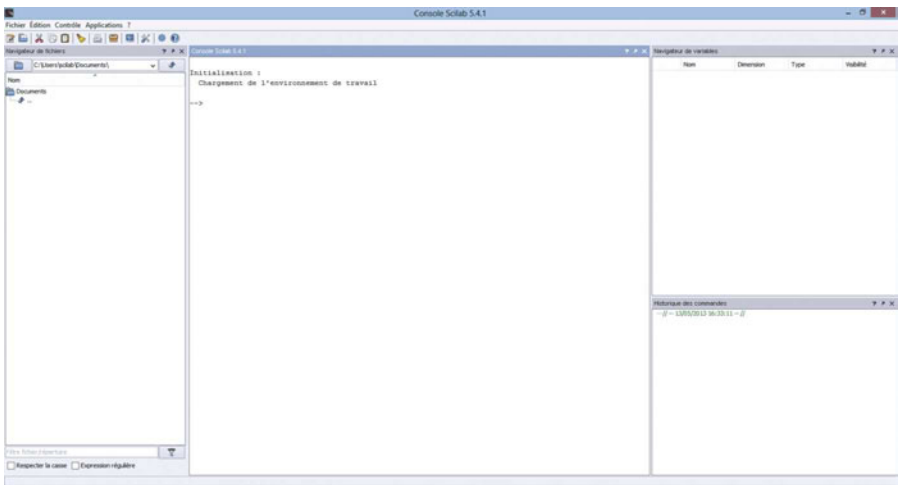
La simulation numérique est aujourd'hui incontournable dans le processus de conception de systèmes. Simuler des phénomènes complexes (physiques, mécaniques, électroniques, etc.) permet d'en étudier les comportements et d'obtenir des résultats sans avoir besoin de recourir à l'expérience réelle. Largement utilisée dans le monde de l'industrie, les ingénieurs et les chercheurs de demain sont formés dès le secondaire aux concepts de modélisation et de simulation.

Xcos est l'outil de Scilab dédié à la modélisation et à la simulation de systèmes dynamiques hybrides incluant à la fois des modèles continus et discrets. Il permet aussi de simuler des systèmes régis par des équations explicites (simulation causale) et implicites (simulation acausale). Xcos inclut un éditeur graphique permettant de représenter facilement des modèles sous forme de schémas fonctionnels (diagrammes) en connectant des blocs entre eux. Chaque bloc représente une fonction de base prédéfinie ou une fonction définie par l'utilisateur.

Distribué librement et gratuitement avec Scilab, Xcos est l'outil idéal pour l'enseignement et l'apprentissage des sciences de l'ingénieur en classe comme à la maison.

L'environnement général

Après avoir lancé Scilab, l'environnement par défaut est constitué d'une console, d'un navigateur de fichiers, d'un navigateur de variables et d'un historique des commandes.



Dans la console, après « -->», il suffit de saisir une commande et d'appuyer sur la touche **Entrée** du clavier pour obtenir le résultat correspondant.

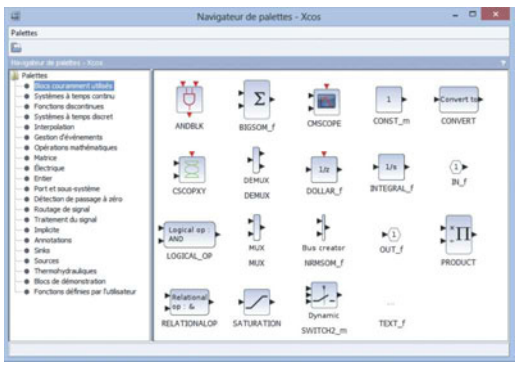
Xcos peut être ouvert :

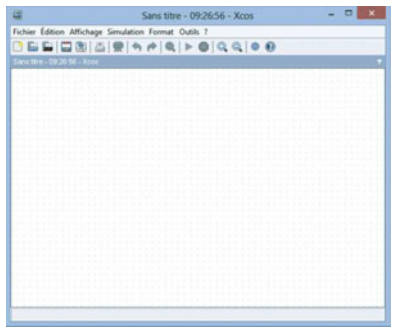
- Depuis la barre d'outils, via l'icône , ou
- Depuis la barre de menus, dans **Applications / Xcos,** ou
- Depuis la console, en tapant :

```
-->xcos
```

Xcos s'ouvre, par défaut, avec deux fenêtres :

- Le navigateur de palettes qui met à disposition un ensemble de blocs prédéfinis,
- Une fenêtre d'édition qui est la zone de construction d'un diagramme.





Pour construire un diagramme, l'utilisateur sélectionne les blocs dans le navigateur de palettes et les positionne dans la fenêtre d'édition (cliquer / glisser / déposer). Il peut ensuite connecter les blocs entre eux en utilisant leurs différents ports (entrée / sortie / événement) pour pouvoir simuler le modèle créé.

La barre de menus

La barre de menus utile dans Xcos est celle de la fenêtre d'édition.

Menu Fichier

- **Nouveau diagramme** (Ctrl+N sous Windows et Linux / Cmd+N sous Mac OS X)

Ouvre une nouvelle fenêtre d'édition de Xcos. Le diagramme Xcos courant n'est pas affecté.

- **Ouvrir** (Ctrl+O sous Windows et Linux / Cmd+O sous Mac OS X)

Charge un fichier Xcos au format .zcos ou .xcos contenant un diagramme ou une palette.

- **Ouvrir le fichier dans le répertoire courant de Scilab**

Charge un fichier Xcos au format .zcos ou .xcos contenant un diagramme ou une palette depuis le répertoire de travail de Scilab.

- **Ouvrir récents**

Propose les fichiers récemment ouverts.

- **Fermer** (Ctrl+W sous Windows et Linux / Cmd+W sous Mac OS X)

Ferme le diagramme courant si plusieurs diagrammes sont ouverts. Quitte Xcos si un seul diagramme est ouvert. Les fenêtres auxiliaires telles que le navigateur de palettes sont également fermées à la fermeture du dernier diagramme.

- **Enregistrer** (Ctrl+S sous Windows et Linux / Cmd+S sous Mac OS X)

Enregistre les modifications apportées à un diagramme. Si celui-ci n'a pas été précédemment enregistré dans un fichier, il sera proposé de l'enregistrer (cf. Enregistrer sous).

- **Enregistrer sous** (Ctrl+Maj+S sous Windows et Linux / Cmd+Maj+S sous Mac OS X)

Enregistre le diagramme ou la palette avec un nouveau nom. Le schéma prend alors le nom du fichier (sans l'extension).

- **Exporter** (Ctrl+E sous Windows et Linux / Cmd+E sous Mac OS X)

Exporte une image du diagramme Xcos courant aux formats standards (PNG, SVG, etc.)

- **Exporter tous les diagrammes**

Exporte des images du diagramme et du contenu de ses superblocs.

- **Imprimer** (Ctrl+P sous Windows et Linux / Cmd+P sous Mac OS X)

Imprime le diagramme courant.

- **Quitter** (Ctrl+Q sous Windows et Linux / Cmd+Q sous Mac OS X)

Quitte Xcos.

Menu Édition

- **Annuler** (Ctrl+Z sous Windows et Linux / Cmd+Z sous Mac OS X)

Annule la ou les dernière(s) opération(s).

- **Rétablir** (Ctrl+Y sous Windows et Linux / Cmd+Y sous Mac OS X)

Rétablit la ou les dernière(s) opération(s) annulée(s).

- **Couper** (Ctrl+X sous Windows et Linux / Cmd+X sous Mac OS X)

Supprime les objets sélectionnés d'un diagramme et en garde une copie dans le presse-papier.

- **Copier** (Ctrl+C sous Windows et Linux / Cmd+C sous Mac OS X)

Place une copie des objets sélectionnés dans le presse-papier.

- **Coller** (Ctrl+V sous Windows et Linux / Cmd+V sous Mac OS X)

Ajoute le contenu du presse-papier au diagramme courant.

- **Supprimer** (Suppr)

Efface les blocs ou les liens qui ont été sélectionnés. Quand un bloc est supprimé, tous les liens qui lui sont connectés sont eux aussi effacés.

- **Tout sélectionner** (Ctrl+A sous Windows et Linux / Cmd+A sous Mac OS X)

Sélectionne tous les éléments du diagramme courant.

- **Inverser la sélection**

Inverse la sélection courante.

- **Paramètres du bloc** (Ctrl+B sous Windows et Linux / Cmd+B sous Mac OS X)

Configure le bloc sélectionné (voir l'aide du bloc pour obtenir plus d'informations sur sa configuration).

- **Zone vers superbloc**

Convertit une sélection de blocs et de liens en un superbloc.

Menu Affichage

- **Zoom avant** (Ctrl+Pavé numérique Plus sous Windows et Linux / Cmd+Pavé numérique Plus sous Mac OS X)

Agrandit la vue de 10 %.

- **Zoom arrière** (Ctrl+Pavé numérique Moins sous Windows et Linux / Cmd+Pavé numérique Moins sous Mac OS X)

Réduit la vue de 10 %.

- **Ajuster le diagramme à la vue**

Ajuste la vue à la taille de la fenêtre.

- **Normal 100 %**

Dimensionne la vue à sa taille par défaut.

- **Navigateur de palettes**

Affiche / masque le navigateur de palettes.

- **Navigateur de diagrammes**

Affiche une fenêtre qui liste les propriétés globales du diagramme et de tous les objets qu'il contient (blocs et liens).

- **Aperçu**

Affiche un aperçu complet du diagramme courant. Avec la vue Aperçu, vous pouvez déplacer l'aire de travail affichée sur une partie du diagramme.

Menu Simulation

- **Configurer**

Modifie les paramètres de simulation.

- **Trace d'exécution et de débogage**

Configure la simulation en mode débogage.

- **Modifier le contexte**

Permet d'entrer des instructions Scilab pour définir des variables / fonctions utilisables dans le paramétrage des blocs d'un diagramme.

- **Compiler**

Compile le diagramme.

- **Initialisation de Modelica**

Permet d'initialiser les variables du sous-ensemble acausal du diagramme.

- **Démarrer**

Lance la simulation.

- **Arrêter**

Interrompt la simulation.

Menu Format

- **Pivoter** (Ctrl+R sous Windows et Linux / Cmd+R sous Mac OS X)

Pivote le ou les bloc(s) sélectionné(s) de 90° anti-horaire.

- **Retourner** (Ctrl+F sous Windows et Linux / Cmd+F sous Mac OS X)

Inverse les positions des entrées et sorties d'événements placées au-dessus et au-dessous d'un bloc sélectionné.

- **Miroir** (Ctrl+M sous Windows et Linux / Cmd+M sous Mac OS X)

Inverse les positions des entrées et sorties régulières placées à gauche et à droite d'un bloc sélectionné.

- **Afficher / Masquer l'ombre**

Affiche / masque l'ombre portée des blocs sélectionnés.

- **Aligner les blocs**

En sélectionnant plusieurs blocs, il est possible de les aligner sur l'axe horizontal (gauche, droite et centre) ou sur l'axe vertical (haut, bas et centre).

- **Couleur de bordure**

Change la couleur des bords des blocs sélectionnés.

- **Couleur de fond**

Change la couleur de remplissage des blocs sélectionnés.

- **Style de liens**

Modifie le style d'un lien.

- **Fond du diagramme**

Change la couleur de fond du diagramme.

- **Grille**

Active / désactive la grille. Avec la grille, le positionnement des blocs et des liens est plus facile.

Menu Outils

- **Génération de code**

Permet de générer le code de simulation associé à un superbloc sélectionné.

Menu ?

- **Aide de Xcos**

Ouvre l'aide sur le fonctionnement de Xcos, des palettes, des blocs et des exemples.

- **Aide du bloc**

Ouvre l'aide sur un bloc sélectionné.

- **Démonstrations Xcos**

Ouvre des exemples de diagrammes et les simule. L'utilisateur peut alors, s'il le souhaite, modifier ces diagrammes et les sauvegarder pour une utilisation future.

Les palettes disponibles

- **Blocs couramment utilisés**

Blocs les plus utilisés.

- **Systèmes à temps continu**

Blocs continus (intégration, dérivée, PID).

- **Fonctions discontinues**

Blocs dont les sorties sont des fonctions discontinues de leurs entrées (hystérésis).

- **Systèmes à temps discret**

Blocs de modélisation en temps discret (dérivée, échantillonné / bloqué).

- **Interpolation**

Blocs calculant des approximations de sortie à partir des entrées.

- **Gestion d'événements**

Blocs permettant de gérer les événements dans le diagramme (horloge, multiplication / division de fréquence).

- **Opérations mathématiques**

Blocs de modélisation des fonctions mathématiques générales (cosinus, sinus, division, multiplication, etc.).

- **Matrice**

Blocs pour des opérations matricielles simples et complexes.

- **Électrique**

Blocs représentant des composants électriques de base (source de tension, résistance, diode, condensateur, etc.).

- **Entier**

Blocs permettant la manipulation de nombres entiers (opérateurs logiques, portes logiques).

- **Port et sous-système**

Blocs de création de sous-systèmes.

- **Détection de passage à zéro**

Blocs utilisés pour détecter les traversées de zéro pendant la simulation. Ces blocs utilisent les capacités des solveurs (ODE ou DAE) pour effectuer cette opération.

- **Routage de signal**

Blocs permettant le routage du signal, multiplexage, aiguillage, échantillonné / bloqué.

- **Traitement du signal**

Blocs pour des applications en traitement du signal.

- **Implicite**

Blocs pour modéliser des systèmes implicites.

- **Annotations**

Blocs utilisés pour les annotations.

- **Sinks**

Blocs de sortie utilisés pour l'affichage graphique (scope) et l'export de données (fichier ou Scilab).

- **Sources**

Blocs de sources de données (impulsion, rampe, sinusoïde) et de lecture de données à partir de fichiers ou de variables Scilab.

- **Thermohydrauliques**

Blocs des composants thermohydrauliques de base (source de pression, tuyaux, vannes de régulation).

- **Blocs de démonstration**

Blocs utilisés dans les diagrammes de démonstration.

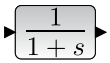
- **Fonctions définies par l'utilisateur**

Blocs utilisateurs permettant de modéliser un comportement (fonction de simulation C, Scilab ou Modelica).

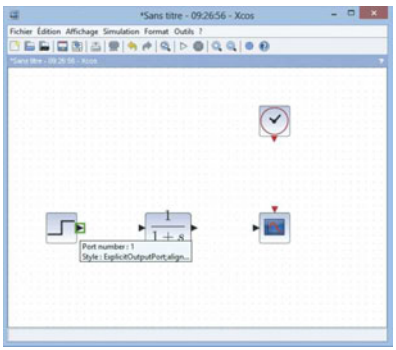
Exemple de construction d'un diagramme simple

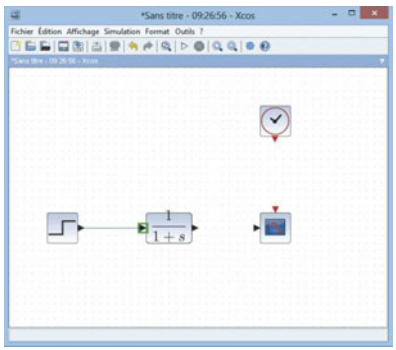
Nous allons vous expliquer comment construire de A à Z, un modèle de système à temps continu modélisé par une fonction de transfert d'ordre 1.

Lancez Xcos.
Comme vu précédemment, Xcos s'ouvre par défaut avec le navigateur de palettes et une fenêtre d'édition.
Dans le navigateur de palettes, nous allons utiliser les blocs suivants :

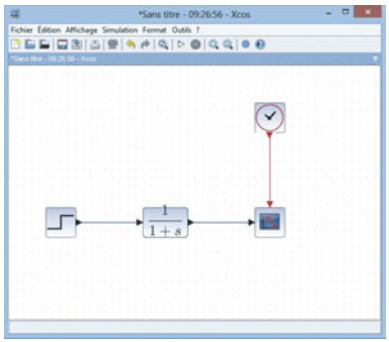
Désignation	Représentation	Sous-palette standard
Échelon		Sources / STEP_FUNCTION
Fonction de transfert continue		Systèmes à temps continu / CLR
Horloge		Sources / CLOCK_C
Visualisation		Sinks / CSCOPE

Disposez les blocs dans la fenêtre d'édition. Pour relier les ports d'entrée et de sortie entre eux, cliquez sur la sortie (flèche noire) du bloc STEP-FUNCTION  et en maintenant le bouton de la souris appuyé, reliez au port d'entrée du bloc CLR, un carré vert apparaît en surbrillance pour indiquer que le lien est correct, comme décrit dans les images ci-dessous :





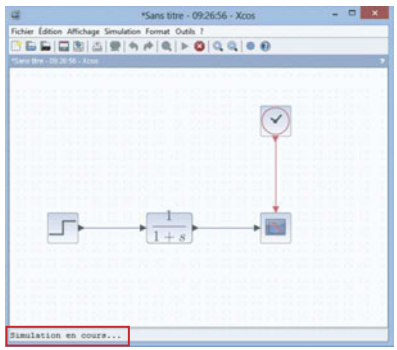
Relâchez pour finaliser le lien.
Complétez ensuite les connexions des blocs entre eux pour arriver à ce résultat :

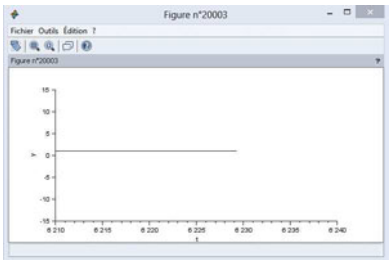


Il est possible d'améliorer l'aspect de votre diagramme en utilisant les options d'alignement des blocs (menu **Format / Aligner les blocs**) et de style de liens (menu **Format / Style de liens**). À tout moment, les blocs peuvent être déplacés ou repositionnés en les sélectionnant et en maintenant le bouton de la souris appuyé pendant le déplacement. Relâchez le bloc à la position souhaitée.

La simulation est lancée en cliquant sur l'icône ▷ (ou depuis le menu **Simulation / Démarrer**) et peut être stoppée en cliquant sur ⊗ (ou depuis le menu **Simulation / Arrêter**).

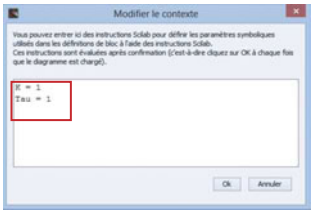
Une nouvelle fenêtre (scope) apparaît, montrant l'évolution de la simulation. En bas de la fenêtre d'édition du diagramme, une mention indique que la simulation est en cours :





Les résultats de cette simulation n'étant pas exploitables, nous choisissons de modifier les paramètres du bloc CLR  et de la simulation.

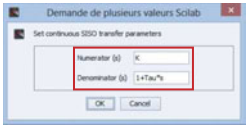
Un « contexte » contenant du script Scilab permet de réutiliser facilement des fonctions et des variables. Nous allons utiliser ce contexte pour fixer des valeurs de référence lors de la simulation du diagramme.



Cliquez sur **Simulation / Modifier le contexte** dans la barre de menus et déclarez les variables suivantes :

- K = 1
- Tau = 1

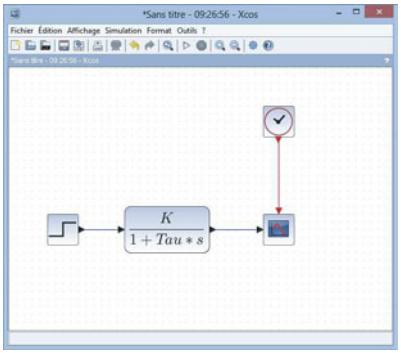
Vous pouvez maintenant utiliser ces variables pour le paramétrage des blocs du diagramme.



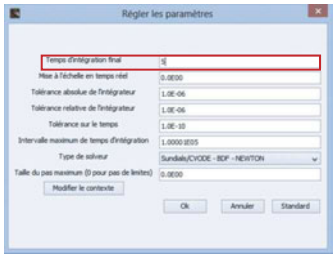
Double-cliquez sur le bloc CLR, une boîte de dialogue s'ouvre avec les paramètres par défaut du bloc. Modifiez ces paramètres :

- Numérateur : K
- Dénominateur : 1+Tau*s

La nouvelle fonction de transfert est affichée sur le bloc :

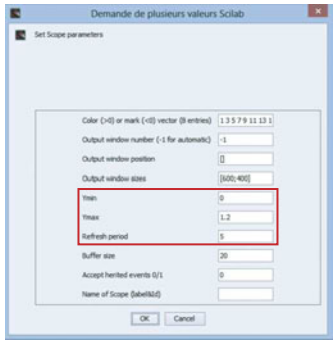


Nous allons maintenant configurer la simulation et les blocs pour visualiser la réponse temporelle du système à une impulsion. Pour cela, nous allons limiter le temps de simulation à 5 secondes (menu **Simulation / Configurer**).

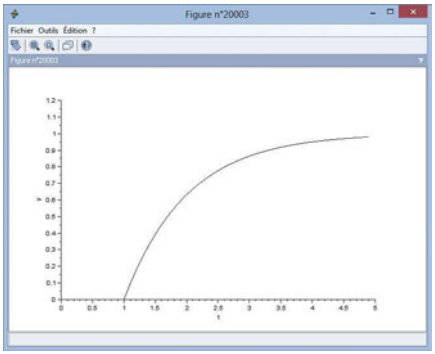


Double-cliquez sur le bloc CSCOPE  pour configurer l'affichage des valeurs comprises entre 0 et 1.2, puis la période de rafraîchissement du scope à 5 secondes. Pour cela, changez les paramètres suivants :

- Ymin : 0
- Ymax : 1.2
- Refresh period : 5

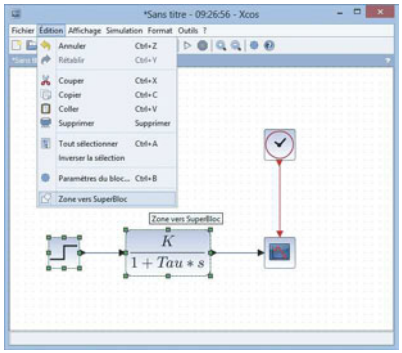


Relancez la simulation et visualisez le résultat :



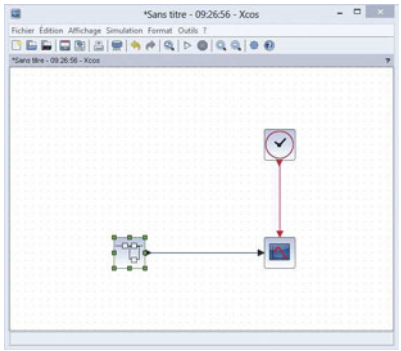
Les superblocs

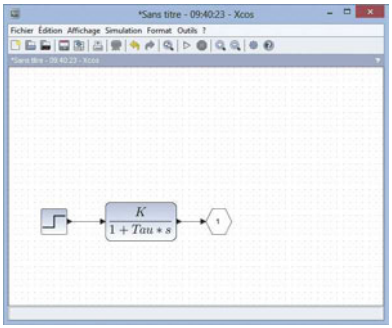
Pour faciliter la compréhension de certains diagrammes, il est souvent utile de recourir aux superblocs ou blocs composites. Un superbloc contient une partie d'un diagramme ainsi que des blocs représentant ses entrées et sorties. Il peut être manipulé comme un seul et unique bloc dans le diagramme parent.



Après avoir réalisé un diagramme et sélectionné la partie du diagramme (ou sous-diagramme) que l'on souhaite réunir en un bloc, la création d'un superbloc s'effectue à partir du menu **Édition / Zone vers Superbloc**.

La sélection est alors devenue un bloc dont on peut afficher le contenu en double-cliquant dessus. Une nouvelle fenêtre d'édition s'ouvre alors avec la sélection de blocs initiale.





Il est également possible de « masquer » le superbloc créé pour désactiver l'accès au sous-diagramme. Pour cela, on effectue un clic-droit sur le superbloc puis **Masque du superbloc / Créer**.
On peut également rendre accessible certains paramètres de configuration du sous-diagramme dans une seule interface de configuration par un clic-droit sur le superbloc, puis **Masque du superbloc / Personnaliser**.
Il suffit ensuite d'ajouter les paramètres que l'on souhaite rendre accessibles.

2 - Modélisation multi-physique acausale (module SIMM)

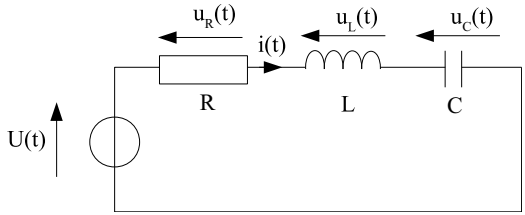
La modélisation dite « acausale » est une forme récente de modélisation des systèmes qui permet de décrire les équations modélisant les phénomènes physiques sans anticiper l'orientation des liens entre les composants ou phénomènes. Il n'y a notamment pas de choix particulier de variables échangées entre composants (force ou vitesse par exemple), ni de notion d'entrée / sortie. Cette particularité conduit à une très grande flexibilité des modèles de composants développés, une réutilisation des modèles sur de nouveaux projets et la possibilité de construire des bibliothèques de composants. Ces avantages en font un outil prisé en entreprise.
D'un point de vue pédagogique, le modèle acausal est très proche de l'architecture matérielle et permet de simuler le comportement d'un système complexe sans avoir à écrire la moindre équation.

L'objectif de cette première partie est de montrer les possibilités de Scilab / Xcos pour simuler des processus multi-physiques par une approche acausale.

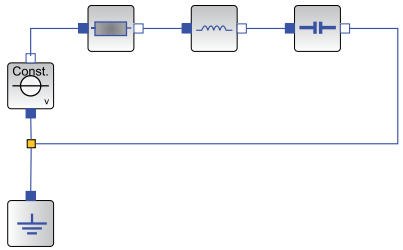
Comparaison des approches causale et acausale

Le circuit RLC alimenté par une source de tension U est représenté par le schéma électrique suivant :

$$u_L(t) = \mathrm{L}\,\frac{di(t)}{dt},\ u_R(t) = \mathrm{R}\,i(t),\ \ i(t) = C\,\frac{du_C(t)}{dt} \text{ et } U(t) = u_R(t) + u_L(t) + u_C(t)$$



La représentation acausale ne privilégie aucune grandeur physique particulière et est basée sur la notion de composants. Ainsi, dans Scilab / Xcos, le diagramme correspondant au circuit RLC est le suivant :



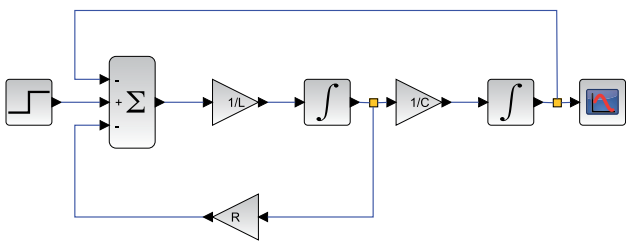
Il n'est pas nécessaire d'écrire les équations du système car chaque bloc les contient. Il suffit simplement de configurer les paramètres de chaque bloc (R, L, C et U) en double-cliquant dessus. Un lien entre deux blocs contient dans ce cas à la fois le potentiel et l'intensité. Cette modélisation permet d'exprimer des relations entre composants sans connaître la grandeur que l'on cherche à calculer. Pour extraire cette grandeur, on utilisera des blocs de type capteur et ainsi entrer dans le monde causal.

Dans une représentation purement causale, les liens représentent une grandeur physique particulière et le diagramme est alors une traduction des équations plutôt qu'une représentation des composants. Lors de la modélisation, la grandeur de sortie est donc exprimée dans une relation directe contenant des blocs « intégrale » pour chacune des équations contenant une dérivée. Cela donne :

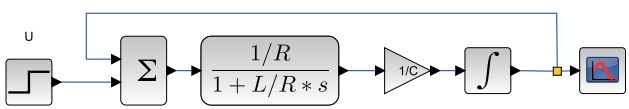
$$u_C(t) = \int \frac{1}{C} . i(t)$$

$$i(t) = \int \frac{1}{L} . (U(t) - u_C(t) - R\, i(t))$$

On peut alors modéliser les équations par un diagramme de ce type :



Celui-ci est donc plus compliqué que le diagramme précédent. On a aussi perdu la signification physique. L'utilisation de la transformée de Laplace permet, entre autre, de simplifier la représentation causale par ce diagramme (en supprimant, par exemple, la boucle avec R) :



Chacune des formes des connecteurs correspond à une grandeur physique. Une validation des liens entre connecteurs est réalisée lors de la simulation afin de garantir une modélisation cohérente.

Reliez les blocs entre eux en respectant obligatoirement les formes et couleurs des connecteurs. Relier un connecteur carré-rouge à un connecteur carré-bleu revient à écrire une égalité entre l'électrique et le thermique sans conversion. Ce type de connexion ne générera pas d'erreur lors de l'édition du diagramme mais une erreur de compilation sera retournée lors de la simulation.

Afin d'éviter les erreurs, veillez à connecter des ports de même domaine fonctionnel :

- Triangle bleu : signal de données (sans dimension).
- Carré bleu : électrique.
- Carré rouge : thermique.
- Carré vert : mécanique 1D en translation.
- Rond gris : mécanique 1D en rotation.
- Carré gris : mécanique 2D plane.

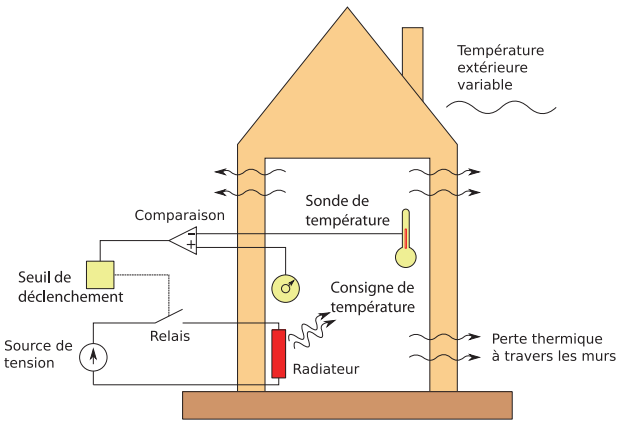
À noter
Certains connecteurs sont vides, d'autres pleins mais il n'y a pas de différences entre eux. Certains composants sont cependant orientés, on repèrera leur polarité en fonction des connecteurs. On rappelle que dans une représentation acausale, il n'y a pas de notion d'entrée / sortie.

Exemple 1 : Régulation de la température intérieure d'une maison d'habitation

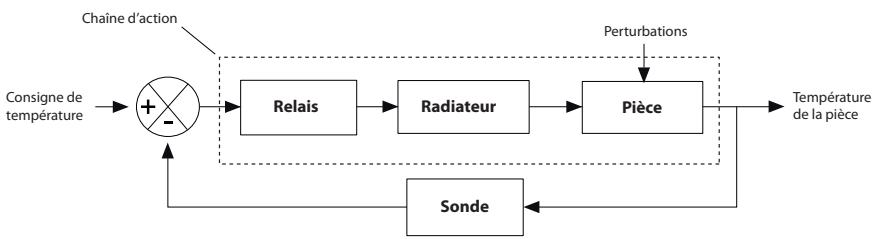
On se propose d'étudier le problème de la régulation de température d'une maison à une température fixe de 20 °C. La température des pièces évolue car la température extérieure passe de 25 °C le jour à 5 °C la nuit et l'isolation thermique n'est pas parfaite.

Les phénomènes physiques sont, en première approximation, assez simples à modéliser :

- L'intérieur de la maison est modélisé par une capacité thermique (ou inertie thermique) qui caractérise la capacité du bâtiment à absorber ou à restituer la chaleur,
- L'isolant thermique de la maison agit comme un conducteur thermique entre l'intérieur et l'extérieur modélisant ainsi les échanges de chaleur,
- Un radiateur chauffe l'intérieur de la maison quand cela est nécessaire et est modélisé par une résistance chauffante.



La commande « Tout-Ou-Rien » (TOR) s'appuie sur la mesure de la température intérieure et la consigne de température de la pièce pour allumer le radiateur quand la température mesurée passe 3 degrés sous la consigne. L'asservissement « Tout-Ou-Rien » est représenté par le schéma fonctionnel suivant :



Comportements physiques élémentaires

La capacité thermique (ou capacité calorifique) d'un corps est une grandeur permettant de quantifier la possibilité qu'a un corps d'absorber ou de restituer de l'énergie par échanges thermiques au cours d'une transformation pendant laquelle sa température varie.

Lancez le logiciel Scilab. Scilab charge automatiquement les modules installés via ATOMS. Lancez ensuite Xcos.

Tous les blocs nécessaires à la simulation de processus se situent dans le navigateur de palettes. Le module SIMM a ajouté un certain nombre de sous-palettes contenant différents blocs. Double-cliquez sur SIMM pour faire apparaître la liste des sous-palettes.

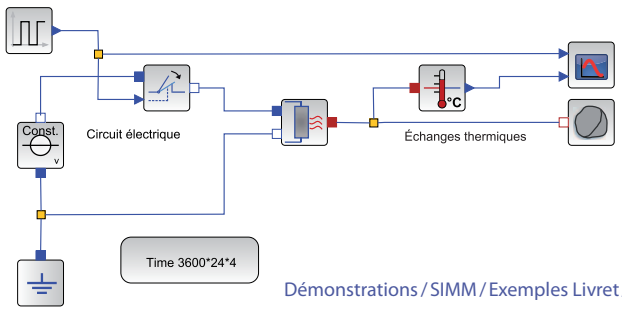
À noter

La capacité thermique est l'énergie qu'il faut apporter à un corps pour augmenter sa température d'un Kelvin. Elle s'exprime en Joule par Kelvin ($J.K^{-1}$). C'est une grandeur extensive. Plus la quantité de matière est importante plus la capacité thermique est grande. La conduction thermique est un transfert thermique spontané d'une région de température élevée vers une région de température plus basse, et est décrite par la loi dite de Fourier $F = -\lambda \frac{dT}{dx}$ où *F* est le flux de chaleur.

Dans SIMM, les composants des chaînes d'énergie et d'information ont été regroupés (sous-palette Composants). On retrouve également une organisation des blocs par domaines (sous-palettes Signaux, Mécanique, Électrique, Thermique).

Modélisation du radiateur et de la maison

On va modéliser le chauffage de la maison par le diagramme suivant :



Démonstrations / SIMM / Exemples Livret / Exemple 1

Positionnez les blocs suivants puis double-cliquez sur chacun d'eux pour configurer les paramètres indiqués lorsque cela est précisé :

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Résistance chauffante		Électrique / Composant basique / Passif / MEAB_HeatingResistor	50 Ω à la température de référence Température de référence de 20 °C = 293.15 K Coefficient de température égal à 0
Interrupteur commandé		Électrique / Composant basique / Passif / MEAI_IdealClosingSwitch	
Source de tension		Électrique / Sources / CEAS_PredefVoltage	Constante 220 V
Masse		Électrique / Sources / MEAB_Ground	
Capacité thermique		Thermique / Basique / MTH_HeatCapacitor	500 000 $J.K^{-1}$

À noter
5.10^5 s'écrit 5e5 dans Scilab.
-4.10^{-3} s'écrit -4e-3 dans Scilab.

Pour changer l'orientation d'un bloc, sélectionnez le puis faire un clic droit et sélectionnez **Format / Pivoter** ou **Format / Miroir** (ou tapez directement Ctrl+R ou Ctrl+M sous Windows et Linux / Cmd+R ou Cmd+M sous Mac OS X).

Pour simuler le comportement du chauffage, on impose en entrée de l'interrupteur commandé, un signal créneau passant de 0 à 1 sur une période de 3 heures (3*3600 secondes car l'unité temporelle est la seconde) et de rapport cyclique 20 %. Ceci signifie que toutes les 3 heures, on chauffe durant 36 minutes. On utilise le bloc MBS_Pulse de la sous-palette Signaux / Sources.

Il faut également spécifier sur quelle durée de simulation le comportement doit être évalué. On choisit une durée de 4 jours. La durée de simulation est spécifiée par le bloc IREP_TEMP (sous-palette Utilitaires / Analyses) où l'on configurera 3600*24*4. On choisit de paramétrer le bloc de manière à ce que 20 000 points de visualisation soient calculés et que les courbes correspondantes soient tracées à la fin du calcul.

Les quantités thermiques, le flux et la température circulent sur le lien reliant la résistance chauffante à la capacité. Pour afficher la température, il faut extraire de ce lien la grandeur température, en utilisant le bloc MTHC_TemperatureSensor de la sous-palette Thermique / Mesure.

La sortie signal (triangle bleu) peut ensuite être visualisée ou utilisée dans le diagramme. Pour visualiser des signaux provenant d'une source ou d'un capteur, on utilise le bloc ISCOPE de la sous-palette Utilitaires / Visualisation pour lequel le nombre d'entrées à visualiser sur un même graphique est spécifié (une légende peut être donnée pour chaque courbe dans le deuxième menu).

 devient  si l'on configure plus d'une entrée (ici 3). Ajoutez donc un bloc ISCOPE à deux entrées pour visualiser le signal créneau et la température de la maison. Indiquez comme nom de courbes « Signal de commande » représentant la consigne et « Température de la pièce ».

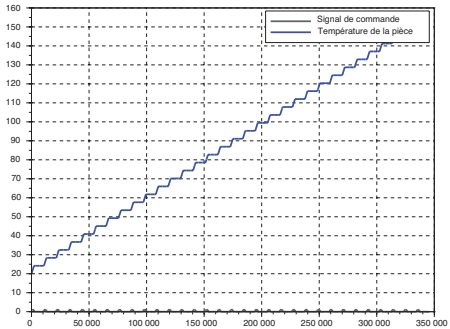
Le tableau ci-dessous résume l'ensemble des blocs à positionner et les paramètres à configurer :

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Signal créneau		Signaux / Sources / MBS_Pulse	Amplitude : 1 Rapport cyclique : 20 % Période : 3*3600 s
Étude temporelle		Utilitiaires / Analyses / IREP_TEMP	Durée : 3600*24*4 s Nombre de points (pour la visualisation et le calcul) : 20 000 Afficher les courbes pendant la simulation : non
Capteur de température		Thermique / Mesure / MTHC_TemperatureSensor	
Visualisation		Utilitaires / Visualisation / ISCOPE	Nombre de courbes : 2

Lancez la simulation.

Les résultats de celle-ci s'affichent sur une nouvelle fenêtre.

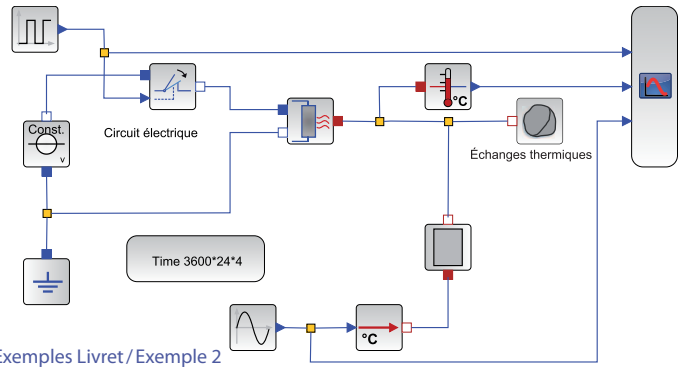
Les courbes montrent notamment qu'un tel modèle, ne prenant pas en compte les pertes au niveau des murs, n'est pas réaliste.



Amélioration du modèle

Prise en compte des pertes

On choisit de prendre en compte les pertes en modélisant le mur par un conducteur thermique et en représentant l'évolution de la température extérieure au cours d'une journée par une sinusoïde.



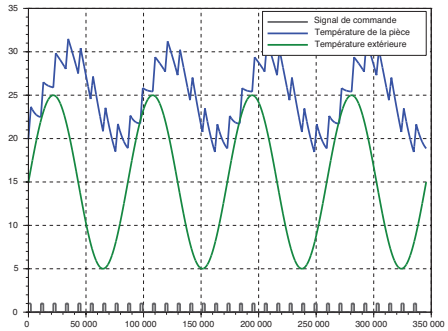
Démonstrations / SIMM / Exemples Livret / Exemple 2

Pour cela, on insère les composants suivants :

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Conductivité thermique		Thermique / Basique / MTH_ThermalConductor	20 $W.K^{-1}$
Source de température	°C	Thermique / Sources / MTHC_PrescribedTemperature	
Signal sinusoïdale		Signaux / Sources / MBS_Sine	Amplitude : 10 Décalage : 15 Fréquence : 1/24h =1/(3600*24) Hz

Le signal sinusoïdal modélise la variation de température extérieure de 5 °C à 25 °C, ce qui justifie les paramètres retenus.

Pour imposer une température comme pour la visualiser, il est nécessaire de spécifier une source de température et d'indiquer sa variation en donnant le signal correspondant. Modifiez le bloc ISCOPE pour ajouter une troisième entrée et visualiser l'évolution de la température extérieure.

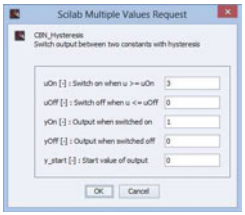


Lancez une nouvelle simulation et observez que la commande par impulsion n'est pas optimale.

Mise en place de la régulation

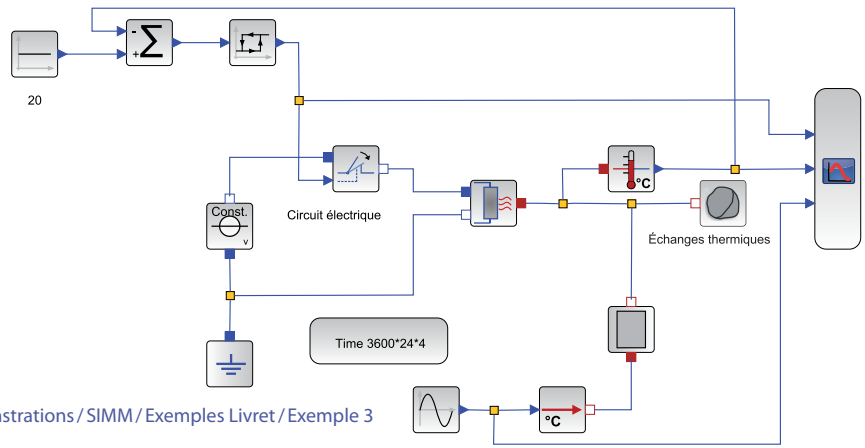
Pour assurer la régulation de température, il suffit de comparer le signal de température de la pièce à une température de consigne. L'écart ainsi formé est utilisé comme entrée d'un relais qui ferme ou non l'interrupteur.

La consigne de température est un bloc signal constant MBS_Constant (sous-palette Signaux / Sources). Une valeur de 20 °C est spécifiée. Pour calculer l'écart, on utilise un bloc sommateur MBM_Add (sous-palette Signaux / Math). Pour changer les signes du sommateur, double-cliquez sur le bloc et renseignez les valeurs -1 et 1.



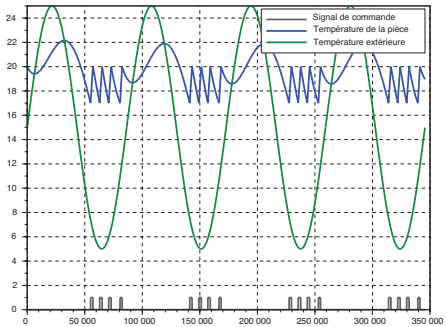
La commande du relais CBN_Hysteresys (sous-palette Signaux / Non-linéarités) passe à 1 (qui correspond à la fermeture) lorsque l'écart est supérieur à 3 °C. Elle repasse à 0 lorsque l'écart est nul. Configurez le bloc comme indiqué ci-contre.

Le diagramme global est alors le suivant :



Démonstrations / SIMM / Exemples Livret / Exemple 3

Relancez la simulation pour obtenir la courbe ci-dessous :



On obtient des résultats de simulation cohérents. Lorsque la température de la maison descend à 17 °C, le système de chauffage se met en route et la température remonte à 20 °C quelle que soit la température extérieure. On constate également que le système ne peut pas refroidir la maison lorsque la température extérieure dépasse la consigne.

À travers cette activité, nous avons pu très facilement voir l'intérêt d'une régulation de température en prenant en compte un modèle très simple d'une maison chauffée par un radiateur. La prise en main du module SIMM est immédiate et ne nécessite que de connaître quelques principes physiques élémentaires.
Cependant, il est nécessaire de garder en mémoire que normalement la tension du courant domestique est sinusoïdale (220 V, 50 Hz) mais évidement, si le simulateur doit représenter correctement le signal sinusoïdal dans le circuit électrique, le pas de temps devra être très petit (de l'ordre de 1 / 500 s) alors que le phénomène thermique est étudié sur plusieurs jours... C'est pourquoi, on utilise une source continue équivalente dans le modèle.

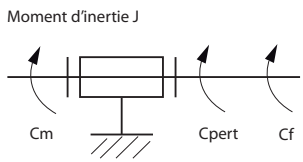
Comme nous le verrons dans l'activité suivante, il faut éviter d'une façon générale d'imposer dans un même modèle des phénomènes dont les constantes de temps sont très éloignées. Cette contrainte conduit à des compromis sur le niveau de détail des modèles choisis et nécessite du recul quant aux phénomènes physiques étudiés.

Exemple 2 : Pilotage d'un moteur à courant continu

La plupart des systèmes développés par les élèves en projet nécessite le pilotage d'un moteur à courant continu. Il est indispensable de modéliser ces moteurs à courant continu pour être capable, par exemple, de réaliser un asservissement ou bien d'évaluer les performances du système. Le moteur à courant continu est modélisé, dans son régime linéaire, par les cinq équations suivantes :

$$U = E + R.I + L.\frac{dI}{dt} \quad (eq1),\ Cm = K.I \quad (eq2),\ E = K.w \quad (eq3),\ Cf = f.w \quad (eq4) \text{ et}$$

$$J.\frac{dw}{dt} = Cm - Cpert - Cf \quad (eq5)$$



- L'équation 1 correspond au modèle électrique du moteur qui est modélisé par une résistance R en série avec une inductance L et une force électromotrice (fem) E.
- Les équations 2 et 3 correspondent aux équations de couplage électromécanique. Le courant circulant dans la bobine génère des forces de Laplace qui se traduisent par un couple moteur Cm.
- L'équation 3 traduit le lien entre la vitesse angulaire et la force électromotrice.
- L'équation 4 décrit le couple dû au frottement visqueux.
- L'équation 5 correspond à l'équation de dynamique obtenue en isolant l'arbre moteur et en lui appliquant le théorème de l'énergie cinétique sachant qu'il est soumis à :
  - un couple moteur Cm,
  - un couple Cpert (frottements secs par exemple)
  - un couple Cf dû au frottement visqueux.

L'objectif de cette activité est de montrer comment modéliser le moteur et le piloter.

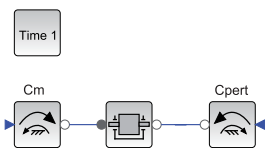
Modélisation mécanique

Le schéma cinématique du moteur soumis à des actions mécaniques peut directement être traduit dans Xcos. Pour commencer, on considère le rotor en liaison pivot soumis à un couple Cm et un couple Cpert. Un solide en rotation autour d'un axe fixe est caractérisé mécaniquement par son moment d'inertie autour de l'axe de rotation (difficulté à le mettre en mouvement).

Positionnez les blocs suivants comme indiqué sur le diagramme ci-après :

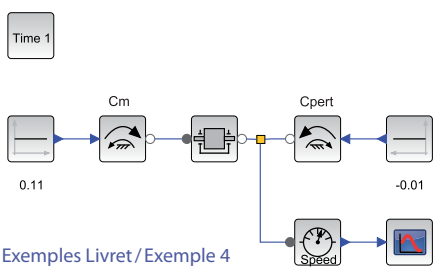
Désignation	Représentation	Sous-palette SIMM	Paramétrage
Solide en rotation autour d'un axe fixe		Mécanique / Rotation 1D / Basique / MMR_Inertia	0.00002 kg.m^2
Couple extérieur (entre l'axe et le bâti)		Mécanique / Rotation 1D / Sources / CMRS_Torque0	
Étude temporelle		Utilitaires / Analyses / IREP_TEMP	Durée : 1 s Nombre de points : 200

On constate que ce diagramme correspond exactement au schéma cinématique. Sur un tel diagramme, on ne spécifie pas les évolutions des grandeurs. De la même manière, il faut indiquer dans Xcos comment évoluent les couples Cm et Cpert.
Reliez ainsi le couple Cm à un signal constant MBS_Constant (sous-palette Signaux / Sources) égal à 0.11 N.m et le couple Cpert à un signal constant de – 0.01 N.m. On peut ensuite visualiser les grandeurs qui transitent sur chaque lien entre les composants. Dans un modèle mécanique 1D de type rotation, ces grandeurs sont l'accélération, la vitesse et la position angulaires ainsi que le couple.



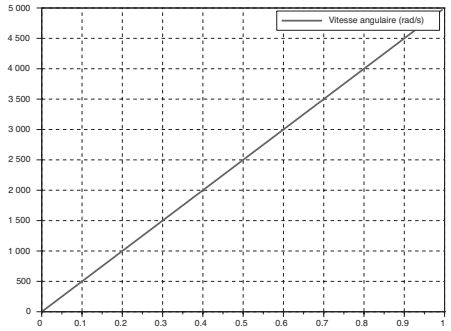
Ajoutez le bloc de mesure CMRS_GenSensor (sous-palette Mécanique / Rotation 1D / Mesure) et double-cliquez dessus. Choisissez de visualiser la vitesse (le texte sur le bloc change). Ajoutez également un bloc ISCOPE (sous-palette Utilitaires / Visualisation) à une entrée.

On obtient le diagramme suivant :



Démonstrations / SIMM / Exemples Livret / Exemple 4

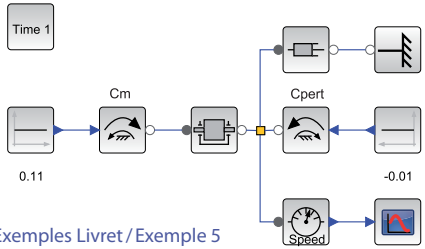
Lancez la simulation et observez une droite de pente 5 000 rad.s^{-1} représentant la vitesse.



Ce résultat permet d'attester le modèle dynamique utilisé et illustre bien la notion d'inertie. Cependant, ce modèle ne prend pas en compte la limitation de l'apport en énergie, ni les frottements.

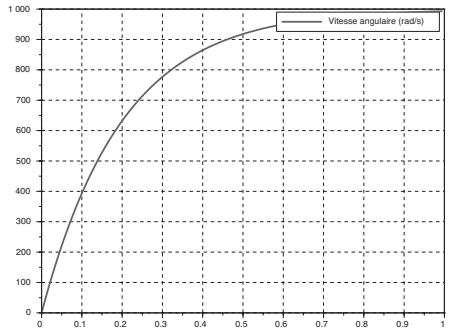
Un couple de frottement visqueux existe entre le bâti et le rotor. Pour le modéliser, il suffit d'ajouter sur le lien représentant l'axe :

- Un bloc amortissement linéaire / visqueux MMR_Damper (sous-palette Mécanique / Rotation 1D / Basique avec une valeur de 0.0001 N.m.s.rad^{-1}),
- Le bâti MMR_Fixed (sous-palette Mécanique / Rotation 1D / Basique).



Démonstrations / SIMM / Exemples Livret / Exemple 5

La simulation sur 1 seconde fournit une réponse d'un système d'ordre 1 tendant vers la valeur (Cm + Cpert) / viscosité avec une constante de temps égale à J / viscosité.



Il ne faut pas oublier de mettre un bâti (référentiel galiléen d'étude) dans une étude mécanique. Il est implicitement spécifié lorsque l'on impose les couples.

À noter
Le comportement en translation rectiligne (Translation 1D) fonctionne exactement sur le même principe.

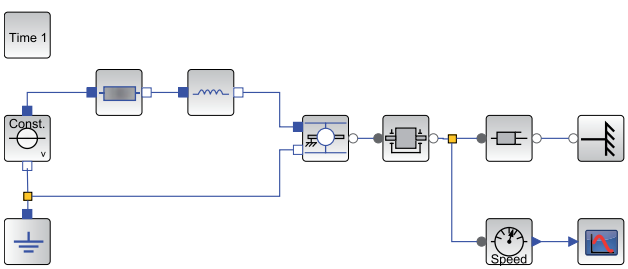
Modélisation électrique et couplage

Positionnez à partir de la sous-palette Électrique, les blocs suivants et reliez-les entre eux :

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Source de tension		Électrique / Sources / CEAS_PredefVoltage	Constante 12 V
Résistance		Électrique / Composant basique / Passif / MEAB_Resistor	1 Ω
Inductance		Électrique / Composant basique / Passif /MEAB_Inductor	0.001 H
Force électromotrice en rotation		Électrique / Composant basique / Passif /CEAB_EMFGEN	0.01 N.m.A^{-1} Relié au bâti (ce qui veut dire que le stator est fixe)
Masse		Électrique / Sources / MEAB_Ground	

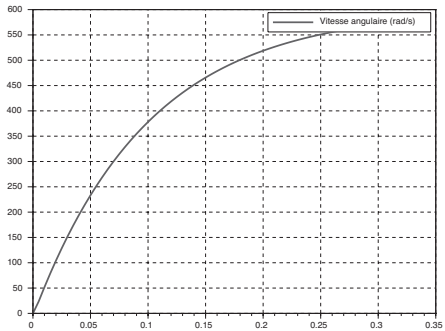
On constate que le bloc CEAB_EMFGEN propose de supposer que le bâti est implicitement donné ou non. Pour la source de tension, les connecteurs ont un sens particulier, le connecteur plein (carré bleu) correspond à la borne positive, le connecteur vide (carré blanc au cadre bleu) à la borne négative.

Le diagramme électrique est à nouveau comparable au circuit électrique. Il ne faut pas oublier de mettre une masse dans le circuit.



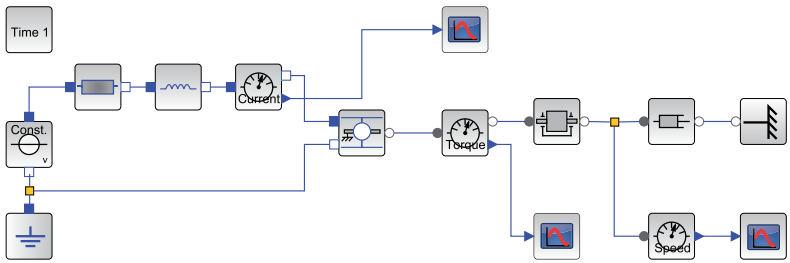
Démonstrations / SIMM / Exemples Livret / Exemple 6

En enlevant le couple résistant (Cpert), on obtient la réponse suivante :



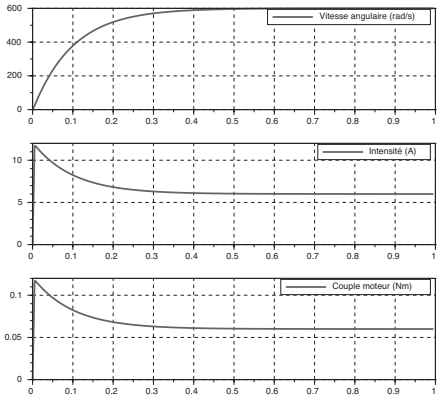
Ajoutez un ampèremètre représenté par le bloc MEAS_CurrentSensor (sous-palette Électrique / Mesure) et placez-le correctement dans le diagramme pour mesurer l'intensité dans le circuit (en série avec l'inductance par exemple). De la même manière que les blocs sources, les bornes positive et négative sont définies par l'aspect plein ou vide des connecteurs.

Positionnez également un capteur de couple MMRS_TorqueSensor (sous-palette Mécanique / Rotation 1D / Mesure) pour mesurer le couple moteur, par analogie avec l'intensité (cf. équation), en série entre la force électromotrice et l'inertie.



Démonstrations / SIMM / Exemples Livret / Exemple 7

On obtient les courbes suivantes :



On constate effectivement que le couple moteur est proportionnel à l'intensité. On peut également étudier l'influence de l'inductance sur la vitesse angulaire en modifiant sa valeur dans le bloc correspondant.

Modélisation du pilotage par hacheur

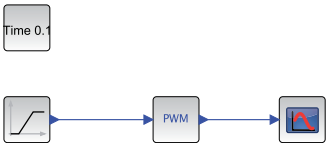
Le hacheur est le préactionneur le plus courant pour l'asservissement des moteurs à courant continu. Il permet, en moyenne, de doser le niveau de tension aux bornes du moteur. Associé à une mesure de courant et une régulation, il peut aussi doser le niveau d'intensité dans le moteur. Le pilotage d'un hacheur se fait par l'intermédiaire d'un PWM (« Pulse Width Modulation » ou MLI, « Modulation de Largeur d'Impulsions »).

Le principe est simple. Un signal créneau de 0 (état logique bas) à 5 V (état logique haut) dont le rapport cyclique, variable, est généré. Comme la fréquence de ce signal est élevée (environ 500 Hz fréquemment), si le système connecté en sortie du PWM est « lent », il ne voit à ses bornes que la tension moyenne du signal PWM (il fonctionne ainsi comme un filtre).

Le hacheur fonctionne selon le même principe que le signal PWM. Il hache la tension issue d'une alimentation externe grâce à des transistors. La tension moyenne dépend alors du rapport cyclique. C'est le signal PWM qui est utilisé pour commander ces transistors.

Signal PWM

Ouvrez une nouvelle fenêtre d'édition Xcos. Positionnez les blocs indiqués pour réaliser le diagramme suivant :



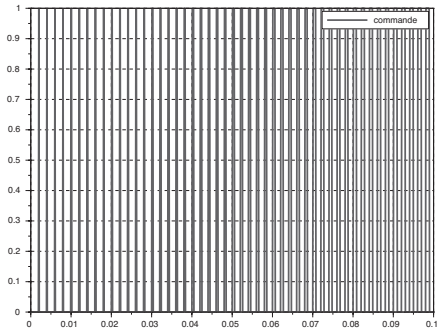
Démonstrations / SIMM / Exemples Livret / Exemple 9

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Echelon		Signaux / Sources / MBS_Step	Amplitude : 125 (le reste à 0)
PWM		Signaux / Sources / CCP_PWM	8 bits Fréquence : 500 Hz Temps de départ : 0 s
Etude temporelle		Utilitaires / Analyse / IREP_TEMP	Durée 0.1s 1000 points Grille : oui Afficher les courbes pendant la simulation : oui
Affichage du signal		Utilitaires / Visualisation / ISCOPE	

Lancez une simulation et observez le signal.

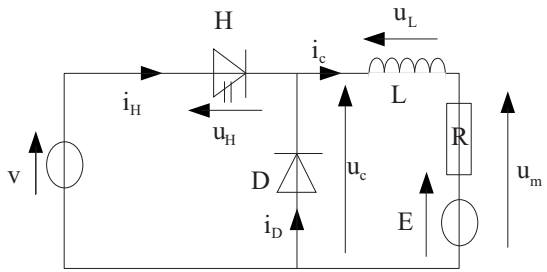
Remplacez l'échelon par un bloc MBS_Ramp (sous-palette Signaux / Sources) de pente 255 / 0.1.

Observez l'évolution du signal créneau en sortie du PWM.

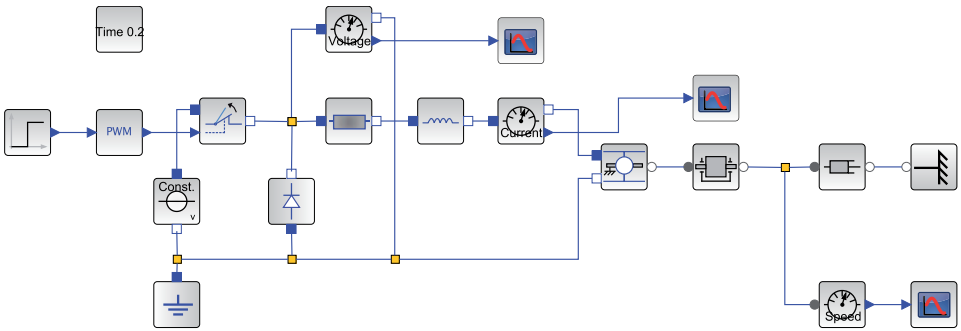


Hacheur 1 quadrant

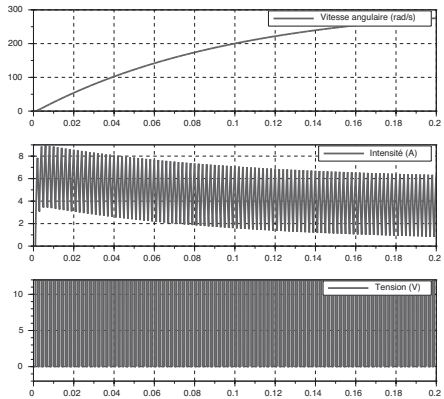
Le hacheur le plus simple est le hacheur série 1 quadrant. Il est constitué d'un transistor commandé (interrupteur commandé) et d'une diode.



Reprenez le diagramme du moteur et ajoutez un interrupteur normalement fermé MEAI_IdealClosingSwitch et une diode CEAI_IdealDiode (sous-palette Électrique / Composant basique / Passif). Insérez le pilotage par PWM donné précédemment en utilisant un échelon d'amplitude 120.



Démonstrations / SIMM / Exemples Livret / Exemple 10



Lancez une simulation sur une durée de 0.2 s (pour 10 000 points).

On obtient par simulation ce que l'on peut observer à l'aide d'un oscilloscope et d'une pince ampèremétrique : une intensité hachée dont l'enveloppe correspond à l'intensité obtenue pour une tension continue de 120 / 256*12 = 5.625 V. La vitesse est par contre naturellement « filtrée ». La tension vue par le moteur est bien un signal créneau de rapport cyclique 120 / 256 = 46.9 %.

Les différents exemples proposés sont relativement simples mais permettent d'illustrer très facilement le comportement électrique et mécanique d'un moteur à courant continu piloté.

On remarque à nouveau qu'il est important de choisir correctement le nombre de points utilisés pour le calcul pour pouvoir observer les phénomènes souhaités. Par exemple, le PWM étant cadencé à 500 Hz, il est nécessaire de prendre au moins 5 000 points pour 1 seconde pour observer son allure. Et même ainsi, en choisissant 1000 points pour 0.2 s, le compilateur ne réussit pas à mener la simulation à bien (l'algorithme de calcul ne converge pas, il faut donc augmenter le nombre de points d'observation de façon conséquente). Comme pour l'étude thermique, il est déraisonnable de modéliser un hacheur pour un moteur ayant une constante de temps bien supérieure à la période du hacheur ou PWM.

Nous verrons dans la prochaine activité qu'il est possible d'utiliser des blocs prédéfinis de certains composants standards (moteurs, hacheurs...) de manière à éviter des diagrammes trop denses.

Découvrez Inria

Aujourd'hui, les technologies numériques rendent les transports plus autonomes et plus sûrs, les maisons plus intelligentes, l'agriculture plus respectueuse de l'environnement... Elles sont à l'origine de nouveaux services, transforment en profondeur nos modes de vie et enrichissent notre quotidien.

Pour se développer, notre société compte toujours plus sur ces technologies numériques qui restent souvent invisibles. Elles sont issues de travaux de recherche longs et complexes associant sciences informatiques et mathématiques.

Créé en 1967, Inria est le seul institut public de recherche entièrement dédié aux sciences du numérique. L'institut réunit aujourd'hui 3500 chercheurs, inventeurs du monde numérique.

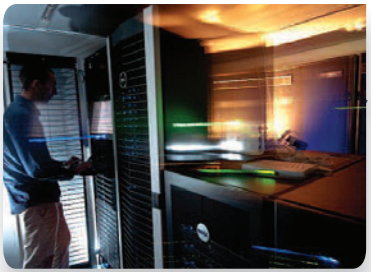
Inria accueille chaque année : **600 stagiaires** de fin d'étude, **plus de 1200 doctorants** dans ses équipes de recherche

Ces chercheurs inventent les technologies numériques de demain.

Issus des plus grandes universités internationales, ils croisent avec créativité recherche fondamentale et recherche appliquée. Ils se consacrent à des problèmes concrets, collaborent avec les acteurs de la recherche publique et privée en France et à l'étranger, **et transfèrent le fruit de leurs travaux vers les entreprises innovantes.**

Et plus d'informations sur :
www.inria.fr
Twitter twitter.com/inria
YouTube youtube.com/inriachannel

Quelques illustrations
de nos recherches

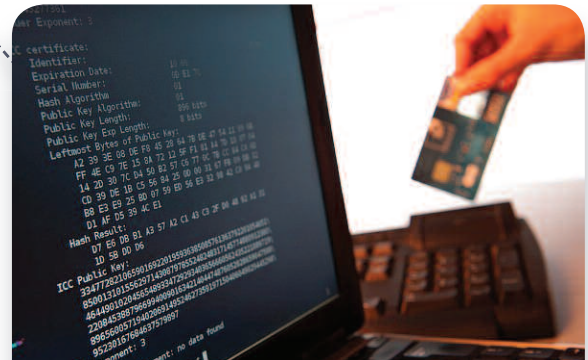


Clusters de calcul pour l'expérimentation
et Télescope (équipes Cartes, Madynes)

Chercheuse (équipe Flowers)

Manipulation moléculaire via écran
multi-touch (équipe Mint)

Extraction des données cryptographiques
d'une carte bancaire



Inria
INVENTEURS DU MONDE NUMÉRIQUE

Exemple 3 : Axe asservi d'angiographie bi-plan

L'étude sera faite sur le déplacement en translation de l'armature suspendue au plafond d'une chaîne image d'un angiographe bi-plan (voir figure ci-contre) permettant la création d'images tridimensionnelles de la structure veineuse d'un patient afin de prévenir les risques de rupture d'anévrisme.

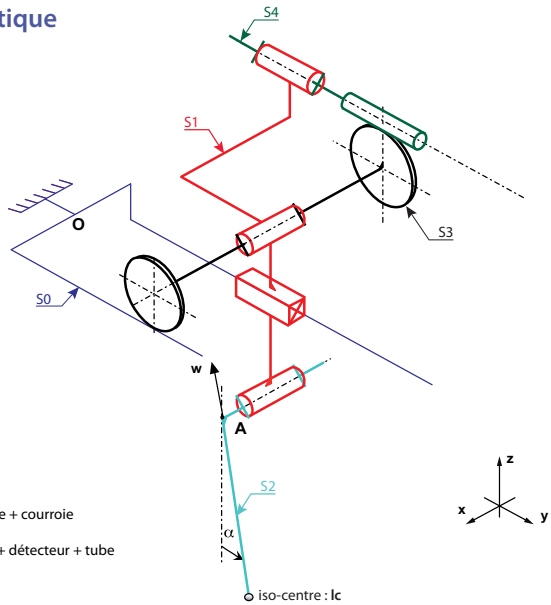
La prise de vue est réalisée en positionnant de manière très précise une tête et un récepteur à rayons X longitudinalement et angulairement, un logiciel dédié se chargeant de la création d'une image colorisée à destination du médecin.

L'ensemble est constitué d'un chariot supportant l'armature (S2) et mis en mouvement par un moteur à courant continu suivi d'un réducteur roue-vis qui entraîne les roues du chariot. Compte-tenu de la masse de l'armature (S2) et de son encombrement, le mécanisme possède deux degrés de liberté : la position du chariot y(t) et la position angulaire de l'armature (S2) α(t) qui apparaît suite à la déformation de l'armature lors du déplacement. On appelle LP (Lateral Plan) l'ensemble constitué de la chaîne image latérale et de son armature suspendue au plafond.

Le mouvement de translation est commandé par le médecin à l'aide d'un joystick. La position voulue étant atteinte, le médecin lâche le joystick et déclenche la prise de vue à l'aide d'une pédale de commande. Le médecin exige que la prise de vue puisse commencer dès la demande d'arrêt du mouvement de translation. Le tableau ci-dessous propose un extrait du cahier des charges.

	Fonction de service	Critère	Niveau
FS1	Déplacer le LP en translation	Être rapide sans mettre en danger le personnel médical	Vitesse maximale 100 mm.s-1
FS2	Pouvoir commencer la prise de vue dès l'arrêt du mouvement de translation	1. Arrêter le mouvement en temps masqué (le temps masqué est lié au temps de réaction du médecin) 2. Limiter l'amplitude et la durée des oscillations du LP au niveau de l'iso-centre	1. Temps masqué : 0.3 s 2. Premières oscillations < à 1 mm d'un extremum à l'autre < à 0.2 mm d'un extremum à l'autre après 1 s
FS3	Assurer la sécurité du patient	Limiter la distance d'arrêt du LP, pour ne pas blesser le patient	Distance d'arrêt réglementée inférieure à 10 mm

Schéma cinématique



S0 : Plafond + rails de guidage + courroie
S1 : Chariot
S2 : Armature en forme de C + détecteur + tube
S3 : Roue + poulie motrice
S4 : Rotor moteur + vis

Données géométriques, cinétiques et grandeurs caractéristiques du moteur

Poulie motrice	Dispositif roue et vis sans fin
Rayon de la poulie motrice S3 : Rp = 0.0318 m	Rapport de réduction : r = 1 / 50
Chariot (S1)	**Armature (S2) : armature en forme de C**
Masse : m_1 = 270 kg	AI_C = d = 1.4 m Centre d'inertie G_2 : AG_2 = l_2 = 0.85 m Masse : m_2 = 490 kg J_2 = 620 kg.m^2
Moteur (inclus dans la masse du chariot S1)	
R = 2.8 Ω (résistance) L = 3.10^{-3} H (inductance) K = 0.23 N.m.A^{-1} (constante de couple ou de fcem) J = 0.25.10^{-3} kg.m^2 (moment d'inertie axe moteur + vis) f = 10^{-5} N.m.s (frottement visqueux)	

Modélisation du système

On s'intéresse dans un premier temps au déplacement de l'ensemble en supposant l'armature fixe par rapport au chariot.

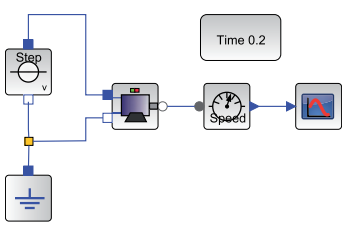
Modélisation du moteur

Nous avons vu dans l'activité précédente qu'il était possible de modéliser de manière fine un moteur à courant continu en utilisant des composants élémentaires. Il est cependant possible de considérer globalement le moteur comme un élément d'une chaîne fonctionnelle et de n'entrer que les caractéristiques de celui-ci sans s'occuper de la manière dont il se comporte.

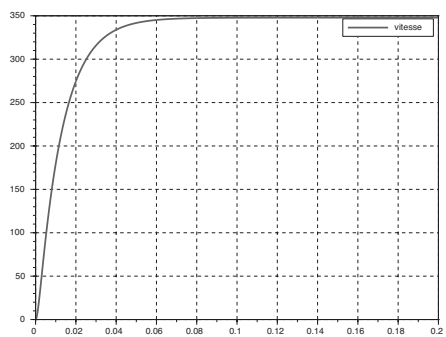
Insérez un bloc MEMC_DCmotor (sous-palette Composants / Actionneurs) dans une fenêtre d'édition. Double-cliquez sur le bloc et configurez les valeurs des constantes du moteur (voir tableau précédent). Ajoutez un bloc CEAS_PredefVoltage (sous-palette Électrique / Sources) et connectez le moteur à cette source d'alimentation. N'oubliez pas d'ajouter une masse MEAB_Ground (sous-palette Électrique / Sources) pour définir le potentiel de référence.

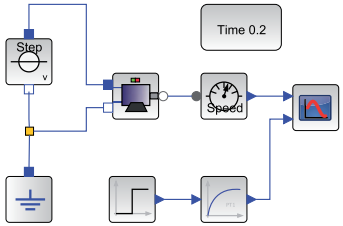
Ajoutez en sortie un bloc CMRS_GenSensor (sous-palette Mécanique / Rotation 1D / Mesure), choisissez de visualiser la vitesse de rotation de l'arbre moteur et ajoutez un bloc ISCOPE (sous-palette Utilitaires / Visualisation).

Nous allons faire différentes études temporelles (bloc IREP_TEMP de la sous-palette Utilitaires / Analyses). Pour commencer, double-cliquez sur le bloc d'alimentation et choisissez un échelon d'amplitude 80 V (tension maximale du moteur). Lancez une simulation sur 0.5 s (en prenant 1000 points) et observez l'allure de la réponse.

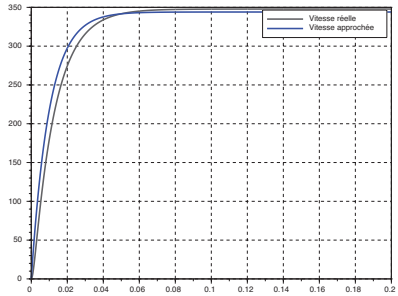


Démonstrations / SIMM / Exemples Livret / Exemple 11





Démonstrations / SIMM / Exemples Livret / Exemple 12



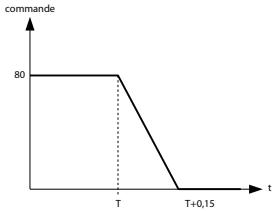
À noter

Le gain du modèle du premier ordre est égal à la valeur asymptotique (environ 350 rad.s^{-1} ici) divisée par la consigne (80 V). La constante de temps est obtenue pour 63 % de la valeur finale (cf. Courbe obtenue pour la charge d'un condensateur).

Une telle réponse correspond à la réponse d'un système du premier ordre de gain et constante de temps donnée.

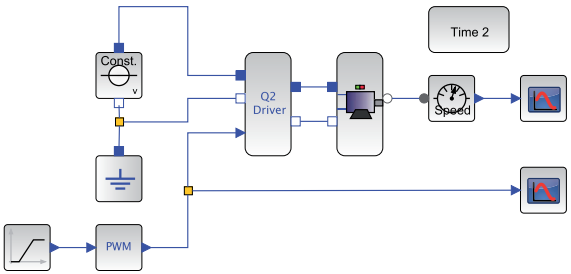
L'avantage de ce modèle est qu'il ne nécessite que deux paramètres alors que le modèle physique en demande beaucoup plus. On peut alors chercher les paramètres qui s'approchent au mieux du modèle du moteur puis les tester avec un bloc MBC_FirstOrder (sous-palette Signaux / Continu) et une source échelon MBS_Step (sous-palette Signaux / Sources) d'amplitude 80.

En réalité, la tension délivrée au moteur ne peut pas être constante car la commande est gérée par un joystick. C'est pourquoi, une commande en trapèze est plus représentative de la commande réelle obtenue lorsque le médecin lâche le joystick. Elle est modélisable par le signal ci-contre.



Pour faire varier la tension d'alimentation du moteur, il est nécessaire d'utiliser un hacheur. Comme pour le moteur, des blocs prédéfinis sont disponibles pour les hacheurs et évitent ainsi d'avoir à détailler leur comportement. Ces composants se situent dans la sous-palette Composants / PréActionneurs. Le hacheur demi-pont MEMC_Q2driver que nous avons choisi pour ce test, est piloté par un PWM ayant une commande sur 8 bits à une fréquence de 500 Hz.

Réalisez le diagramme indiqué sur la page suivante et paramétrez le signal de consigne du PWM pour obtenir une tension de commande du moteur comme indiqué sur la figure précédente (on prendra T = 2 s et 20 000 points de simulation). Ajoutez un bloc pour visualiser le signal du PWM.



Démonstrations / SIMM / Exemples Livret / Exemple 13

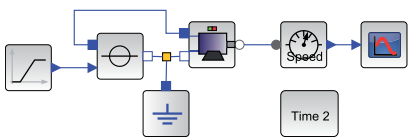
Les blocs à ajouter ou paramétrer à nouveau sont rappelés ci-dessous :

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Ramp		Signaux / Sources / MBS_Ramp	Amplitude : -255 Durée : 0.15 s Décalage : 255 Temps de début : 1 s
PWM	PWM	Signaux / Sources / CCP_PWM	8 bits Fréquence : 500 Hz Temps de début : 0 s
Étude temporelle	Time 100	Utilitaires / Analyses / IREP_TEMP	Durée : 2 s 10 000 points Grille affichée : oui Afficher les courbes pendant la simulation : non
Affichage du signal		Utilitaires / Visualisation / ISCOPE	Nombre de courbes : 1 Nom de la courbe : vitesse PWM
Hacheur demi-pont	Q2driver	Composants / PréActionneurs / MEMC_Q2driver	

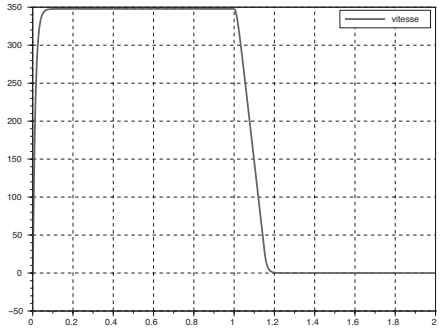
On constate qu'il est très difficile de visualiser correctement le signal PWM (ou en sortie du hacheur) sur la durée de simulation compte-tenu de la commande retenue.

Ouvrez une nouvelle fenêtre d'édition Xcos et copiez le diagramme précédent. Supprimez les blocs hacheurs, PWM, source d'alimentation et remplacez-les par une alimentation variable pilotée MEAS_SignalVoltage (sous-palette Électrique / Sources). Modifiez le bloc Trapèze MBS_Ramp (sous-palette Signaux / Sources) pour obtenir directement la tension de commande du moteur (Amplitude -80, Décalage 80 V).

Lancez la simulation en prenant 2 000 points. Comparez la courbe obtenue avec la précédente.



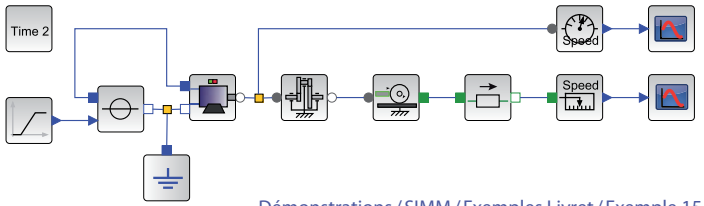
Démonstrations / SIMM / Exemples Livret / Exemple 14



En observant les résultat obtenus, on s'aperçoit que la modélisation fine du hacheur n'est pas nécessaire dans ce cas précis. Il sera remplacé par la source de tension variable.

Modélisation des adaptateurs

En aval du moteur à courant continu étudié précédemment se trouve une chaîne cinématique composée par la mise en série d'un réducteur de type roue et vis sans fin. La rotation de la roue du réducteur entraîne la translation du chariot par l'intermédiaire d'une roue de rayon donné.



Démonstrations / SIMM / Exemples Livret / Exemple 15

Ajoutez un réducteur à engrenages MMR_IdealGearGen et un système de transformation de mouvement de rotation en translation MMR_IdealGearR2TGen (sous-palette Composants / Adaptateurs). Configurez les blocs à l'aide des données initiales. Attention pour le bloc engrenage, le rapport renseigné est le rapport de l'entrée sur la sortie. Intercalez juste avant le capteur de vitesse linéaire un bloc CMTC_Mass (masse en translation) de la sous-palette Mécanique / Translation 1D / Basique. Seule la masse est importante (760 Kg). Il est également possible de définir les positions et vitesses initiales, etc.

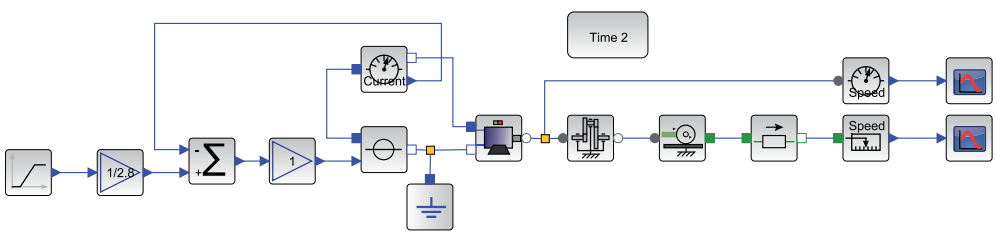
Lancez ensuite une simulation. Vérifiez que la vitesse obtenue est cohérente.

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Réducteur à engrenages		Composants / Adaptateurs / MMR_IdealGearGen	Rapport de transmission : 50 Bâti : oui
Pignon crémaillère		Composants / Adaptateurs / MMR_IdealGearR2TGen	Rapport de transmission : 1 / 0.038 Bâti : oui
Masse en translation		Mécanique / Translation 1D / Basique / CMTC_Mass	Masse : 760 Kg
Capteur de vitesse		Mécanique / Translation 1D / Mesure / CMTS_GenSensor	Vitesse

Influence des frottements sur l'évolution de l'intensité

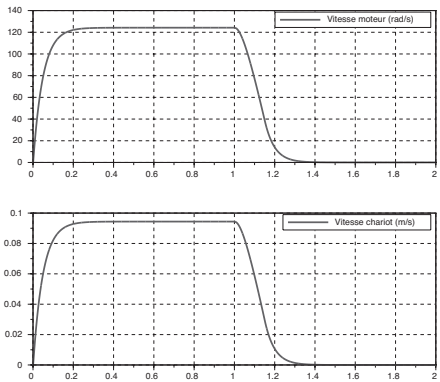
Il existe une résistance à l'avancement qui est égale à 70 N.
Cette action peut être modélisée par l'intermédiaire du bloc CMT_MassWithFriction (sous-palette Mécanique / Translation 1D / Basique). Ce bloc est plus compliqué à paramétrer car il permet de tester différents modèles de frottement. Le seul qui est considéré ici est le modèle de Coulomb (les autres sont mis à 0). Attention, le signe de la résistance est positif.

On constate que la commande utilisée ne respecte pas la vitesse maximale de 0.1 $m.s^{-1}$. Modifiez celle-ci de manière à ce que la vitesse maximale soit celle définie dans le cahier des charges.
Ajoutez un ampèremètre au niveau de l'alimentation du moteur et visualisez l'intensité. Faites varier la masse et visualisez l'influence sur l'intensité (ajoutez une masse sans frottement s'il y a des problèmes de simulation).



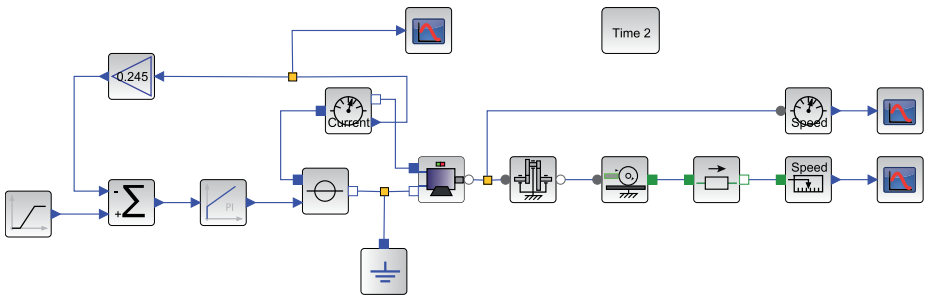
Démonstrations / SIMM / Exemples Livret / Exemple 16

On constate ainsi qu'il est nécessaire de contrôler l'intensité du moteur pour gérer les accélérations de l'armature. C'est pourquoi par la suite, un asservissement d'intensité est mis en place, comme on le voit sur la figure suivante avec une correction unitaire.



Mise en place de l'asservissement d'intensité du moteur

Insérez une boucle de courant (comme pour la régulation de température) en prenant un correcteur de type PI proportionnel intégral MBC_PI (sous-palette Signaux / Continu). En pratique, la mesure de courant est réalisée par une résistance ou une sonde ampèremétrique de gain égal à 0.245 $V.A^{-1}$.



Démonstrations / SIMM / Exemples Livret / Exemple 17

L'asservissement portant sur le couple, il faut imposer une intensité de consigne, soit U / R avec U la valeur maximale utilisée précédemment (vous pouvez aussi ajouter un gain comme sur la figure précédente).

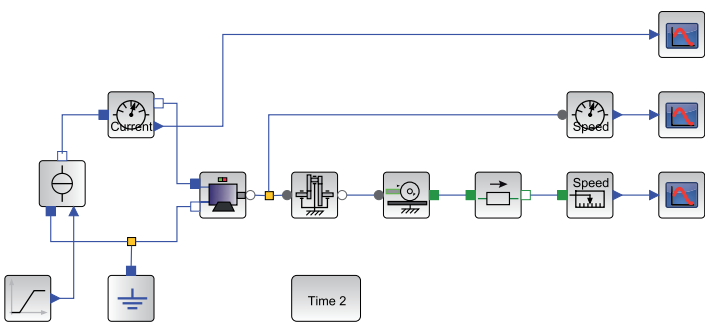
Le réglage du correcteur PI pour l'asservissement de courant du moteur se fait en choisissant une constante de temps égale à 10*L / R. Renseignez cette valeur et laissez un gain proportionnel de 1 pour l'instant.
Remarquez que la vitesse n'est plus du tout la même car le moteur est piloté en intensité (donc à une intensité constante correspondra une vitesse linéaire de par l'équation de dynamique). Il est donc indispensable d'adapter la consigne d'intensité pour retrouver des niveaux de vitesse raisonnable. Changez la consigne (prendre 1.7 V par exemple ou 0.6 A). Observez ensuite l'influence du gain du correcteur sur l'intensité.

Le choix final de la valeur du gain doit se faire en fonction de la tension du moteur qui ne peut pas dépasser physiquement 80 V (Cf. hacheur). Ajoutez un voltmètre aux bornes du moteur et visualisez la tension pour différents gains supérieurs à 1. Une valeur trop grande du gain entraînera nécessairement un dépassement de la tension maximale. On prendra par la suite un gain de 100.

Asservissement en vitesse

La partie précédente a permis de montrer qu'il est nécessaire d'asservir en intensité un moteur pour contrôler les accélérations. Il est donc indispensable d'ajouter un contrôle de vitesse pour assurer la loi de consigne souhaitée.

Pour simplifier la commande du moteur, on peut utiliser une source d'intensité pilotée MEAS_SignalCurrent (sous-palette Électrique / Sources) plutôt que l'asservissement d'intensité, ce qui revient à considérer cet asservissement comme parfait.



Démonstrations / SIMM / Exemples Livret / Exemple 18

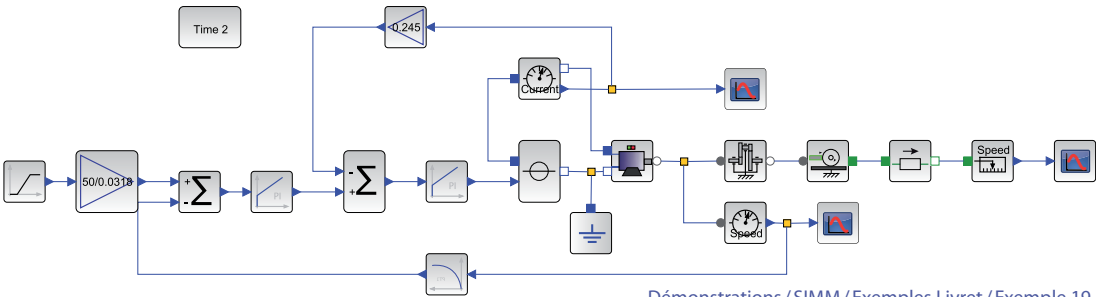
L'asservissement de vitesse se fait en comparant la vitesse de rotation du moteur (capteur sur l'axe du moteur) à la vitesse de consigne (en trapèze). On utilise à nouveau un correcteur PI pour simplifier (gain 0.15, constante de temps 0.025 s).

Le codeur incrémental monté sur l'axe moteur délivre une position en nombre de tops (informations binaires) et cette information de position est ensuite « dérivée » numériquement par différences finies. Le signal obtenu, image de la vitesse de rotation du moteur, est en général très bruité. Il faut donc adjoindre à cette mesure un filtre de constante de temps donné. Ce filtre est modélisable simplement par un bloc du premier ordre MBC_FirstOrder (sous-palette Signaux / Continu) de constante de temps 1 ms.

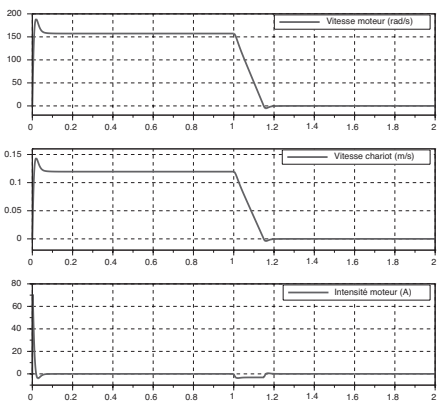
Réalisez l'asservissement de vitesse suivant et configurez en entrée une consigne rampe de valeur maximale 0.1 $m.s^{-1}$. Lancez la simulation et vous pouvez maintenant conclure sur la pertinence de ce type de commande.

Les blocs à ajouter ou paramétrer sont rappelés ci-dessous :

Désignation	Représentation	Sous-palette SIMM	Paramétrage
Ramp		Signaux / Sources / MBS_Ramp	Amplitude : - 0.1 Durée : 0.15 s Décalage : 0.1 Temps de départ : 1 s
Gain		Signaux / Math / MBM_Gain	50/0.0318
Étude temporelle		Utilitaires / Analyses / IREP_TEMP	Durée : 2 s 2000 points Grille affichée : oui Afficher les courbes pendant la simulation : non
Comparateur		Signaux / Math / MBM_Add	Gains 1 et -1
Correcteur PI		Signaux / Continu / MBC_PI	Gain 0.15 Constante de temps 0.025 s
Filtre du premier ordre		Signaux / Continu / MBC_FirstOrder	Gain 1 Constante de temps 0.001 s



Démonstrations / SIMM / Exemples Livret / Exemple 19



On a montré à travers cette application la nécessité d'asservir en intensité (ou en couple) le moteur de manière à contrôler les accélérations. On parle, dans ces conditions, de pilotage en intensité du moteur. Il est alors indispensable d'asservir en vitesse le moteur pour obtenir un mouvement particulier du chariot respectant ainsi le cahier des charges initial.

De nombreux axes linéaires (imprimante par exemple) utilisent ce type de commande (boucle d'intensité et de vitesse) pour contrôler précisément les mouvements. La suite de cette activité serait de modéliser les oscillations de l'armature en utilisant la sous-palette Mécanique / Plane et d'améliorer la commande pour supprimer les problèmes de résonance.

3- Modélisation et analyse de systèmes à temps continu (Module CPGE)

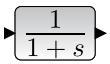
L'exemple que nous allons traiter est le même que celui utilisé dans le chapitre précédent : l'angiographe. Le module CPGE permet de dimensionner les correcteurs à mettre en place dans le système par des analyses fréquentielles. Il nécessite des connaissances théoriques (transformée de Laplace, analyse fréquentielle, correcteur...) pour être utilisée efficacement. Son utilisation est donc plutôt orientée post-bac.

Le cahier des charges du système est donc le même que dans le chapitre précédent, page 42.

Mise en place d'un diagramme de moteur à courant continu

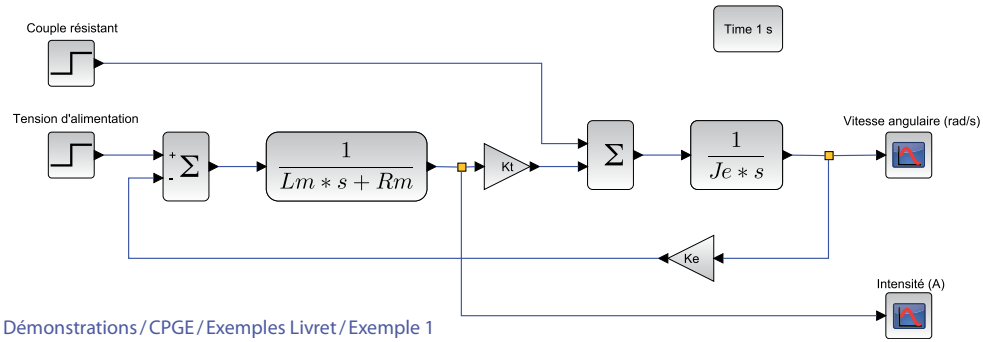
Dans cette première partie, nous allons construire le diagramme du moteur à courant continu (résistance Rm, inductance Lm, constante de couple Kt, constante de vitesse Ke et inertie équivalente rapportée à l'axe moteur Je).

Ouvrez une nouvelle fenêtre d'édition Xcos. Positionnez les blocs suivants :

Désignation	Représentation	Sous-palette CPGE	Paramétrage
Échelon		Entrées / STEP_FUNCTION	Valeur finale : 0
			Valeur finale : 80
Fonction de transfert continue		Opérateurs linéaires / CLR	Numérateur : 1 Dénominateur : Lm*s + Rm
Gain		Opérateurs linéaires / GAINBLK_f	Kt
			Ke
Comparateur		Opérateurs linéaires / BIGSOM_f	
Visualisation		Sorties / SCOPE	Intensité (A) Vitesse angulaire (rad / s)

À noter
Pour configurer un sommateur en soustracteur, cliquez sur le bloc et choisissez la forme du vecteur [1,-1] ou 1 -1 (sans crochets, ni virgule). Il est possible de sommer ou de soustraire plus de deux valeurs en augmentant la taille du vecteur (exemple : 1 1 1 pour la somme de 3 entrées).

Construire le modèle du diagramme du moteur à courant continu en reliant les différents blocs (après les avoir positionnés puis éventuellement orientés), sous la forme du diagramme ci-dessous :



Démonstrations / CPGE / Exemples Livret / Exemple 1

Pour définir les grandeurs de sortie (afin que les courbes obtenues soient ensuite repérables facilement), double-cliquez sur chacun des deux blocs SCOPE et configurez le nombre de courbes à superposer sur un même graphe (une seule dans notre cas) puis après avoir cliqué sur OK, entrez le nom du signal, à savoir une vitesse de rotation de l'axe du moteur (en rad.s^{-1}) et l'intensité dans l'induit (en A).

En double-cliquant sur un espace vierge du diagramme, il est possible d'insérer du texte permettant d'améliorer la lecture globale. Dans l'image précédente, on a ainsi pu décrire à quoi correspondaient les deux échelons d'entrée et les deux sorties. Il est également possible d'attacher une étiquette / texte à un bloc par un simple clic droit sur le bloc, puis **Format / Édition** et de compléter la zone de texte.

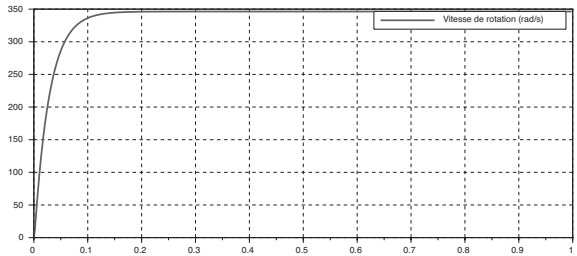
À noter
Dans toutes les zones de texte, il est possible d'entrer du code LaTeX permettant, par exemple, d'insérer des équations.

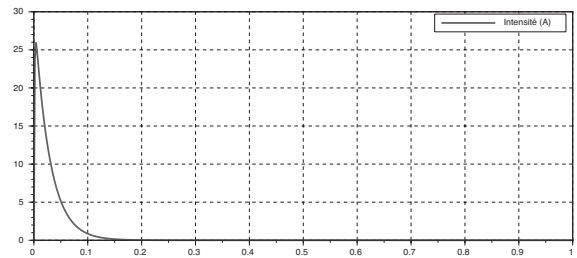
En double-cliquant successivement sur les deux échelons d'entrée, configurez une tension de 80 V (tension nominale du moteur), un instant initial t = 0 s et un couple résistant nul à t = 0 s (le cas perturbé sera étudié par la suite).

Étude / Simulation

Pour lancer une simulation, il est nécessaire de spécifier le type d'étude retenu (temporelle et / ou fréquentielle).

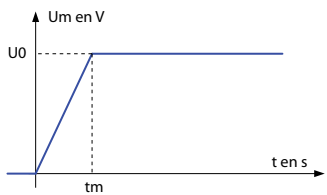
Pour réaliser une étude temporelle, positionnez dans le diagramme un bloc REP_TEMP (sous-palette Analyses). En double-cliquant sur ce bloc, configurez une durée de simulation de 1 s et 500 points d'affichage. Lancez alors la simulation. Deux courbes s'affichent, représentant respectivement l'intensité et la vitesse angulaire.





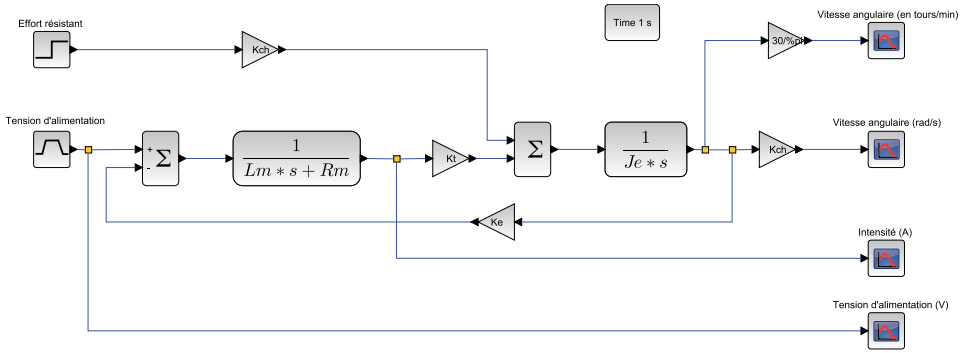
MISE EN PLACE D'UN DIAGRAMME DE COMMANDE EN BOUCLE OUVERTE

En aval du moteur à courant continu se trouve une chaîne cinématique (réducteur à engrenages + système roue et vis sans fin + système pignon – crémaillère) de rapport de réduction Kch = $31.8.10^{-3}$ / 50 $m.rad^{-1}$ permettant de transformer la rotation en une translation de l'axe linéaire. Afin de ne pas trop solliciter le moteur électrique, on l'alimente progressivement jusqu'à la valeur nominale (voir figure ci-contre) : on donne tm = 0.1 s (temps de montée) et U0 = 40 V (tension nominale). Ce signal peut classiquement être obtenu par soustraction d'une rampe de pente U0 / tm à l'instant t = 0 s et d'une rampe de même pente mais décalée dans le temps de tm secondes. Cependant, le module CPGE offre la possibilité de définir un signal trapèze dont nous n'exploiterons que la partie initiale (montée et maintien) en prenant un temps de maintien en position maximale très grand. Par ailleurs, un effort Fr = – 72 N est exercé sur la structure qui se translate à l'instant tr = 0.5 s. Cet effort est ressenti comme un couple résistant au niveau du moteur avec, si l'on suppose un rendement unitaire, un rapport de proportionnalité correspondant au gain de la chaîne cinématique.



Complétez le contexte avec les informations Kch = $31.8.10^{-3}$ / 50, tm, tr, U0 et Fr à la suite des précédentes puis ajoutez trois gains GAINBLK_f (sous-palette Opérateurs linéaires). Rajoutez également deux afficheurs SCOPE (sous-palette Sorties) pour le tracé de la tension d'alimentation et de la vitesse de déplacement. Remplacez l'échelon de tension par un bloc TRAPEZOID (sous-palette Entrées) et configurez l'amplitude, le temps de montée et prenez un temps de maintien de 10 (donc très supérieur au temps de simulation) sans modifier les autres paramètres.

À partir du diagramme initial, créez alors le diagramme ci-dessous, correspondant à la commande en boucle ouverte du système (entrée : tension du moteur en V, sortie : vitesse de déplacement de l'axe en $m.s^{-1}$ perturbation : force Fr à l'instant tr = 0.5 s). Un gain de 30 / π (qui s'écrit dans Scilab 30 / %pi) permet d'obtenir la vitesse de rotation en $rad.s^{-1}$.

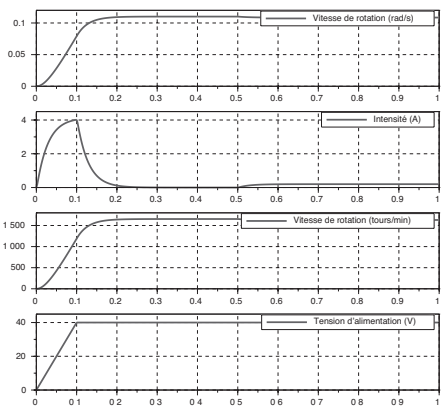


Démonstrations / CPGE / Exemples Livret / Exemple 2

Lancez la simulation.

Quatre courbes s'affichent, représentant respectivement la tension d'alimentation en V, l'intensité dans l'induit en A, la vitesse angulaire en tours.min^{-1} et en $rad.s^{-1}$.

Analysez rapidement les courbes obtenues en les comparant à celles qui ont été obtenues par la simulation acausale.



PRÉSENTATION DE LA STRUCTURE DE L'ASSERVISSEMENT EN VITESSE

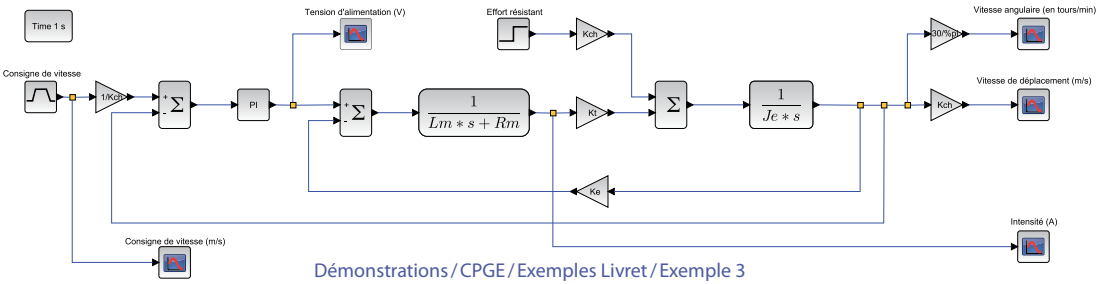
La structure de commande précédente n'est pas performante car elle est très sensible à la perturbation. Afin d'arrêter le déplacement de l'axe au niveau souhaité, il serait nécessaire de couper l'alimentation au bon moment (donc en tenant compte à la fois des perturbations et de l'inertie du système), ce qui est bien entendu illusoire.

Pour maîtriser la vitesse de déplacement de l'axe et compenser les perturbations, il est donc nécessaire de passer à une structure asservie. Dans le cas du système étudié, ceci est réalisé par l'implantation d'un codeur incrémental sur l'axe moteur et l'adjonction d'un calculateur permettant de traiter les informations de manière numérique.

Nous introduisons les nouveaux blocs suivants :

Désignation	Représentation	Sous-palette CPGE	Paramétrage
Générateur de signal trapézoïdal		Entrées / TRAPEZOID	Amplitude : 0.1 Largeur : 10
Correcteur PI		Opérateurs linéaires / PIcontrol	Gain proportionnel : Kp

La structure du diagramme est la suivante :



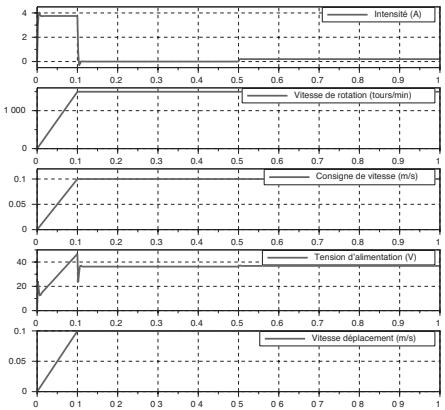
Démonstrations / CPGE / Exemples Livret / Exemple 3

Dans ce diagramme, on note les évolutions suivantes :

- Modification de la typologie de l'entrée pour réaliser une commande en trapèze de vitesse (valeur de maintien de 0.1 $m.s^{-1}$ au bout de 0.1 s), ce qui correspond à une commande classique pour ce type d'axe linéaire asservi,

- Ajout d'une observation de l'évolution de la tension d'alimentation du moteur,
- Mise en place d'un soustracteur et d'un correcteur PI, PIcontrol (sous-palette Opérateurs linéaires).

Modifiez le contexte en définissant la variable Kp=1. On utilise pour l'instant comme paramètres du correcteur PI, le gain proportionnel Kp et un gain intégral nul. Cette variable tient compte de l'ensemble correcteur + module d'amplification (hacheur) en amont de l'ensemble constitué du moteur et de la chaîne cinématique.



ASSERVISSEMENT DE VITESSE ET CORRECTION PROPORTIONNELLE

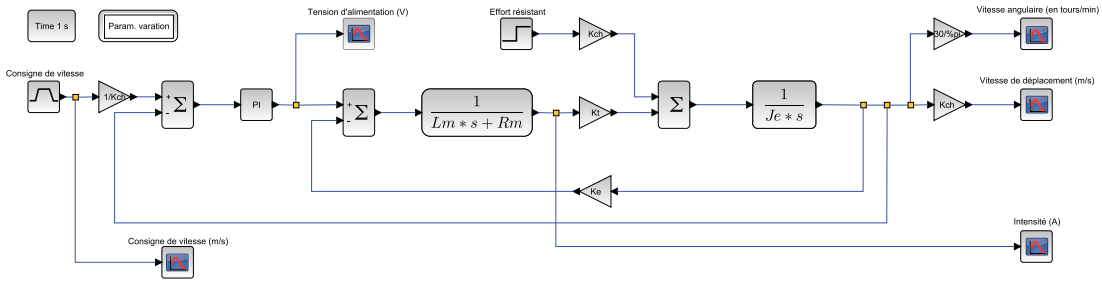
Nous allons, dans un premier temps analyser l'influence de la simple correction proportionnelle implantée sur le comportement temporel du système. Dans le contexte, la valeur de Kp est unitaire. La valeur n'a pas d'importance mais, pour être prise en compte, elle doit être définie.

Nous introduisons les nouveaux blocs suivants :

Désignation	Représentation	Sous-palette CPGE	Paramétrage
Variation paramétrique	Param. varation	Analyses / PARAM_VAR	Kp [0.5,1,5,10,100]
Saturation		Non-linéarités / SATURATION	40 -40

Insérez un bloc PARAM_VAR (sous-palette Analyses) pour faire une étude paramétrique.

À noter
Le bloc PARAM_VAR permet de faire des études paramétriques sans avoir besoin de linéariser le système. Le diagramme peut donc utiliser n'importe quel type de blocs.

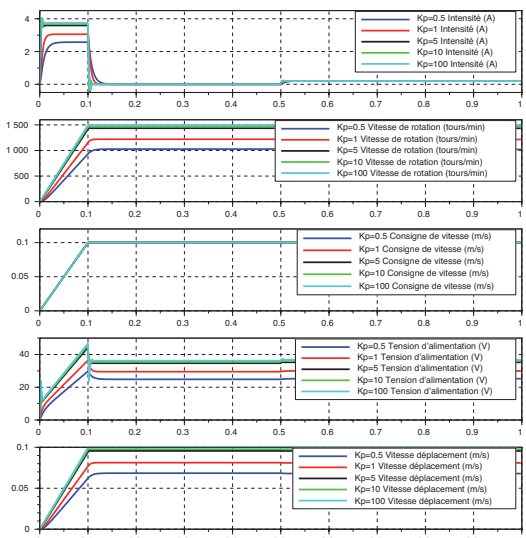


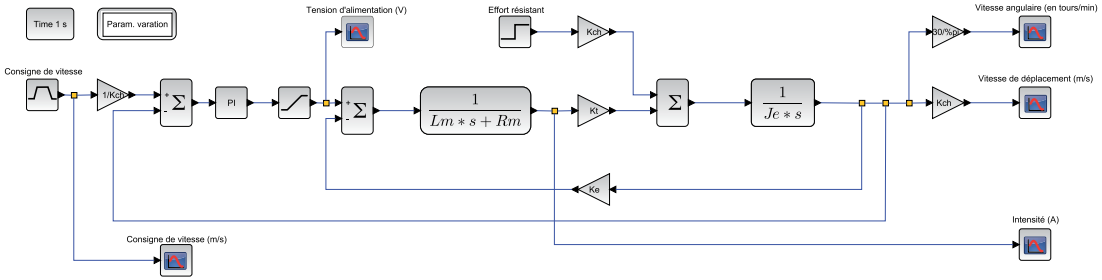
Lancez la simulation et observez l'influence du gain du correcteur sur les performances en comparaison aux attentes du cahier des charges.

La valeur de la tension d'alimentation doit physiquement être limitée à 40 V pour protéger le moteur. On constate cependant que, pour plusieurs valeurs de Kp, cette tension est dépassée. Pour améliorer le modèle, il est donc indispensable d'ajouter une non-linéarité physique : la saturation.

Sélectionnez tous les blocs situés après le correcteur (les blocs sont mis en surbrillance) et déplacez un des blocs pour mouvoir l'ensemble. Insérez alors, entre le bloc correspondant à l'ensemble correcteur + hacheur et le moteur, un bloc SATURATION (sous-palette Non-linéarités).

Configurez la saturation avec les valeurs indiquées dans le tableau précédent (40 et -40).



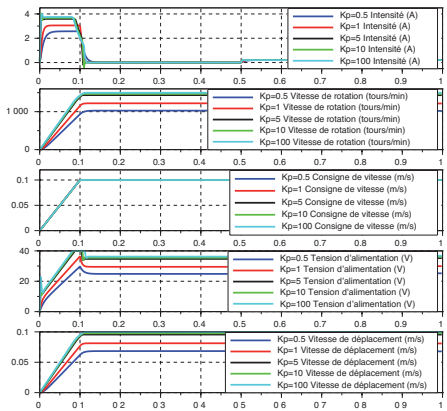


Relancez la simulation et observez l'effet de la saturation sur la réponse obtenue, tant au niveau des grandeurs électriques que sur le suivi de la consigne de vitesse en trapèze.

Afin d'analyser la capacité du système à suivre l'évolution de la consigne, il est bien entendu possible de comparer la réponse temporelle obtenue à la consigne en superposant plusieurs courbes sur un même afficheur (SCOPE).

Double-cliquez sur le bloc SCOPE de la vitesse de déplacement et demandez deux courbes de noms « Vitesse de déplacement ($m.s^{-1}$) » et « Consigne de vitesse ($m.s^{-1}$) » puis connectez la nouvelle entrée.

Lancez la simulation et observez l'effet de la saturation pour la valeur Kp = 100.



ASSERVISSEMENT DE VITESSE ET CORRECTION PROPORTIONNELLE ET INTÉGRALE

Afin d'améliorer sensiblement la capacité du système à suivre la consigne de vitesse, il est nécessaire d'apporter un effet intégral à la correction.
On choisit alors de renseigner le gain Ki du correcteur PI : $C(p) = Kp + \frac{Ki}{p} = Kp\left(1 + \frac{1}{Ti\,p}\right)$

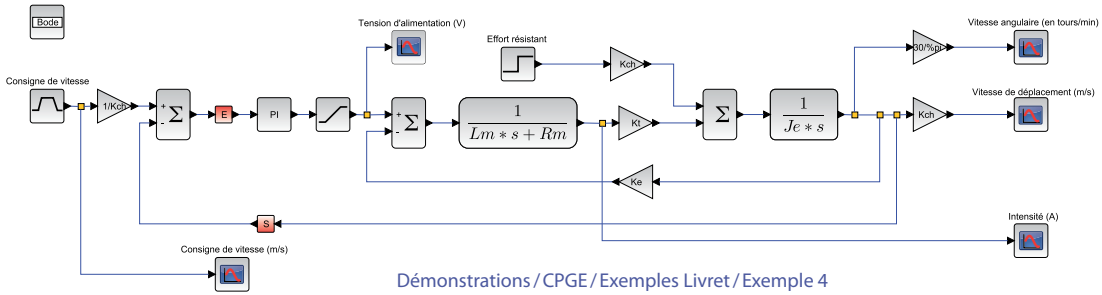
Pour analyser l'influence des paramètres et le réglage du correcteur, on réalise une analyse fréquentielle en boucle ouverte. Pour réaliser cette analyse, il est nécessaire de linéariser le diagramme. Normalement les blocs non-linéaires (type saturation) sont remplacés par des gains unitaires (sauf pour le retard qui peut être traité exactement en boucle ouverte).

Désignation	Représentation	Sous-palette CPGE	Paramétrage
Analyse fréquentielle	Bode	Analyses / REP_FREQ	
Points d'entrée / sortie	Nom	Analyses / GRANDEUR_PHYSIQUE	

Supprimez le bloc REP_TEMP et remplacez-le par un bloc REP_FREQ (sous-palette Analyses) qui réalisera l'analyse fréquentielle. On peut cumuler la réponse temporelle et la réponse fréquentielle en laissant les deux blocs REP_ TEMP et REP_FREQ sur le schéma.

Ajoutez également deux blocs GRANDEUR_PHYSIQUE (sous-palette Analyses) qui sont utilisés pour définir les points d'entrée et de sortie de l'analyse fréquentielle. Double-cliquez sur ces blocs pour les nommer « E » et « S » (tout nom est possible). Positionnez-les avant le bloc PIcontrol et au niveau du retour du premier comparateur (attention à bien les relier).

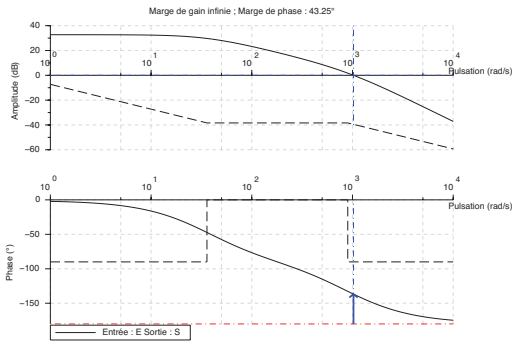
On doit alors se trouver dans la situation du diagramme suivant :



Démonstrations / CPGE / Exemples Livret / Exemple 4

À noter
Plusieurs blocs REP_FREQ peuvent être introduits dans le diagramme, ce qui permet d'obtenir la réponse fréquentielle de la boucle ouverte et celle de la boucle fermée dans deux fenêtres graphiques séparées. Si l'on souhaite superposer deux réponses fréquentielles, il suffit de configurer les différentes entrées et sorties séparées par des points-virgules. Exemple : E1;E2 et S1;S2.

On peut alors configurer, dans le bloc REP_FREQ, le type de diagramme fréquentiel (Bode, Black ou Nyquist), les points d'entrée et de sortie (référence aux blocs GRANDEUR_PHYSIQUE « E » et « S ») ainsi que l'affichage ou non des marges de stabilité et des asymptotes. Dans la fenêtre de configuration du bloc, il est possible de définir les pulsations minimale et maximale : prendre ici 1 et 10 000.
Lancez une simulation (analyse fréquentielle seule) pour les paramètres par défaut du correcteur (Kp=1 dans le contexte et Ki=0). Visualisez alors les marges de gain et phase sur les diagrammes de Bode.



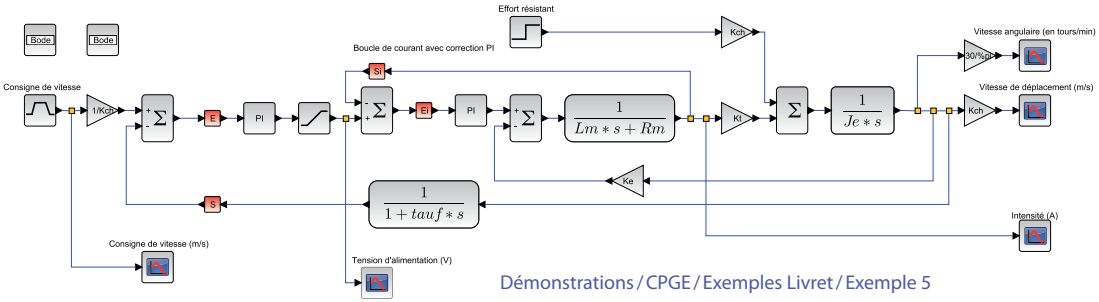
Modifiez les valeurs pour voir leur impact sur le comportement fréquentiel de la boucle ouverte. Il est nécessaire de relancer une simulation à chaque fois. Vous pouvez également positionner un bloc PARAM_VAR pour voir l'influence des paramètres pour des jeux donnés en analyse fréquentielle. Attention, le calcul de marges n'est fait que pour une seule courbe (pas le tracé des asymptotes).

Un réglage satisfaisant pour avoir une marge de phase de 45 ° environ est Kp=10 et Ki=0.1.

Ajoutez le bloc REP_TEMP (durée 1 s, 200 points de tracés) et observez la réponse temporelle. Vous pouvez supprimer le bloc REP_FREQ pour n'observer que la réponse temporelle.

On voit que ce réglage PI n'est pas idéal dans le cas d'étude car le moteur utilisé peine à entraîner de manière efficace le système avec cette structure série. Afin d'améliorer sensiblement les performances, on commande le moteur en courant c'est-à-dire que l'on ajoute une boucle de courant. Cette structure de commande est très classique de nos jours. La quasi-totalité des cartes de commande actuelles dispose de cette fonctionnalité que nous utilisons ici.

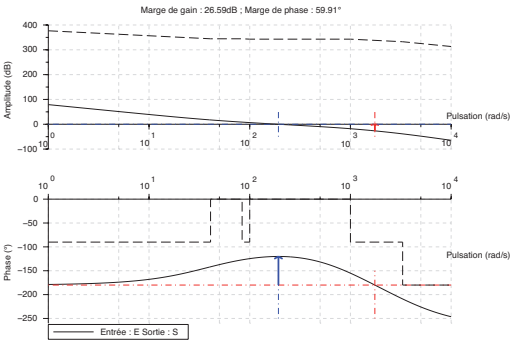
Le principe de la boucle de courant est simple. On vient, grâce à une structure adéquate, mesurer l'intensité au niveau de l'induit et alors modifier l'alimentation réelle du moteur après une correction le plus souvent de type PI comme le montre le diagramme suivant :

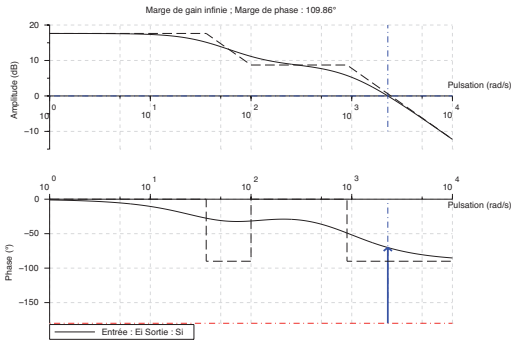


Démonstrations / CPGE / Exemples Livret / Exemple 5

Dans ce diagramme, le retour de courant a été choisi unitaire car le gain du capteur (résistance de shunt ou capteur à effet Hall) est ici intégré dans le correcteur PI.
Il est nécessaire de régler le correcteur PI de la boucle de courant (utilisant les grandeurs physiques Ei et Si) puis le correcteur PI de la boucle de vitesse. Un filtre a été ajouté dans la boucle de retour tachymétrique pour être au plus prêt de la physique du système (mesure par codeur puis dérivation numérique bruitée).

On voit qu'il est possible, en utilisant cette structure, d'atteindre des performances relativement importantes tout en ne modifiant ni la structure globale de l'asservissement (donc en gardant un codeur incrémental sur l'axe du moteur) ni le choix du moteur à courant continu, ce qui est toujours délicat (coût, implantation, etc.).

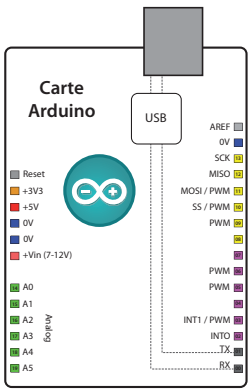




4- Acquisition et pilotage de moteur (Module Arduino)

Cette activité expose les possibilités de simulation de type « hardware in the loop » (intégration de matériel physique dans une simulation logicielle) de Scilab / Xcos et du module Arduino couplés à une carte Arduino Uno. Le module Arduino permet, en effet, d'intégrer dans l'outil de simulation, la commande d'une carte Arduino Uno afin de faire de l'acquisition ou du pilotage de systèmes. Il est possible de réaliser les commandes suivantes : lecture / écriture d'une entrée / sortie logique, lecture / écriture d'une entrée / sortie analogique, commande d'un moteur à courant continu, d'un servomoteur et d'un moteur pas à pas.

Présentation succincte de la carte Arduino Uno



Sur la carte Arduino Uno, identifiez les ports (PINS en anglais) suivants :

- **14 Entrées / Sorties logiques** (port ou « PIN Digital » de 0 à 13) :
  - Série asynchrone (avec 0 sur Rx et 1 sur Tx). Les PINS 0 et 1 ne seront donc pas utilisables,
  - 2 interruptions externes sur 2 et 3 (utilisées pour le codeur en quadrature),
  - Sortie 13 couplée à une LED sur la carte.
- **6 Entrées analogiques** (A0 à A5) :
  - La tension d'entrée doit nécessairement être inférieure à la tension de référence (5 V ou 1.1 V ou AREF : référence externe),
  - 6 CAN (Convertisseur Analogique Numérique) 10 bits (plage de 1024) à 10 kHz maximum,
  - Ces entrées peuvent aussi fonctionner comme des Entrées / Sorties numériques.
- **6 Sorties analogiques** : 6 PWM sur les ports 3, 5, 6, 9, 10 et 11, construites sur les pins d'entrées / sorties logiques.

La programmation de toutes les cartes de la famille Arduino, dont le modèle Uno, se fait dans un langage simplifié adapté du C / C++, basé sur l'utilisation de « classes » (ou macro-commandes) faciles à comprendre et à modifier, y compris par des personnes non-spécialistes de la programmation. Le projet « Open Source » Arduino a, en effet, initialement été créé pour la mise en œuvre d'environnements sonores ou visuels à destination de créations artistiques. Il était donc indispensable que l'interface soit réduite à la mise en œuvre séquentielle d'ordres simples et à l'acquisition de données analogiques ou numériques. Le logiciel de programmation, gratuit et utilisable sans installation sur l'ordinateur sur les environnements Windows, Mac OS X et Linux, est téléchargeable en ligne à l'adresse : http://arduino.cc/en/Main/Software
Un guide d'installation est disponible à l'adresse :
http://arduino.cc/en/Guide/HomePage

À noter
Comme les mémoires sont de type Flash, le programme reste « indéfiniment » en mémoire, même sans alimentation, après son implantation dans le microcontrôleur. Pour charger le programme, la liaison USB permet de communiquer avec le microcontrôleur en émulant une liaison série.

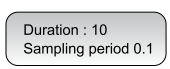
UTILISATION DU MODULE ARDUINO INTÉGRÉ À SCILAB / XCOS

Pour faire fonctionner le module, il faut aussi charger un programme particulier (Toolbox_Arduino.ino) dans l'Arduino. Celui-ci est téléchargeable depuis le site http://atoms.scilab.org/toolboxes/arduino.
Grâce au logiciel Arduino, ouvrez le fichier téléchargé puis téléversez-le après avoir choisi le port de communication dans le menu Outils / Port série (prendre le port différent du Com1) et le type de carte Arduino Uno. Vérifiez que le téléversement s'est bien terminé et quittez.
La carte Arduino Uno est prête à être interfacée avec Scilab / Xcos grâce au module que nous allons maintenant étudier. Il ne sera plus nécessaire d'utiliser le logiciel Arduino par la suite.

Démarrage avec le module Arduino : clignotement d'une LED et entrées / sorties logiques

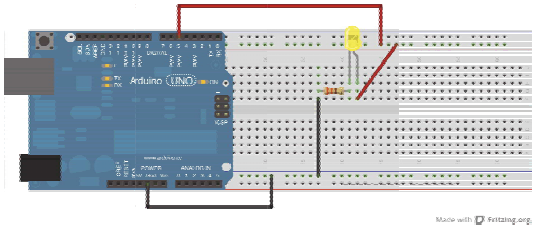
Lancez Xcos. Pour vérifier le bon fonctionnement du module, nous allons faire clignoter la LED intégrée à la carte Arduino (PIN 13), ce qui correspond à l'exemple de base en programmation de microcontrôleurs.

Le tableau ci-dessous résume l'ensemble des blocs à positionner et les paramètres à configurer :

Désignation	Représentation	Sous-palette	Paramétrage
Configuration Arduino		Arduino / Configuration / ARDUINO_SETUP	Port série : 5
Configuration simulation		Arduino / Configuration / TIME_SAMPLE	
Écriture port digital		Arduino / Digital / DIGITAL_WRITE_SB	PIN digital : 13
Générateur d'impulsions		CPGE / Entrées / PULSE_SC	Retard : 0.1 Largeur de pulsations : 30 Période : 1 Amplitude : 1

Reliez l'entrée Pulse et la sortie logique puis lancez l'acquisition via le bouton de simulation. Les deux diodes Rx et Tx sur la carte doivent indiquer la communication série entre Xcos et l'Arduino tandis que la LED située à côté du PIN 13 doit clignoter.

Vous pouvez également prendre une LED quelconque, lui associer une résistance judicieusement choisie et brancher une patte de la LED + résistance au PIN Gnd (« ground » pour masse) et l'autre extrémité à un PIN digital quelconque (différent de 0 et 1 - 5 sur le schéma ci-contre), puis configurer ce numéro de PIN dans le bloc DIGITAL_WRITE_SB du diagramme.



À l'issue de ce premier essai, l'interfaçage des entrées / sorties logiques de l'Arduino avec Xcos a été mis en œuvre. On peut maintenant passer à l'acquisition d'une grandeur analogique.

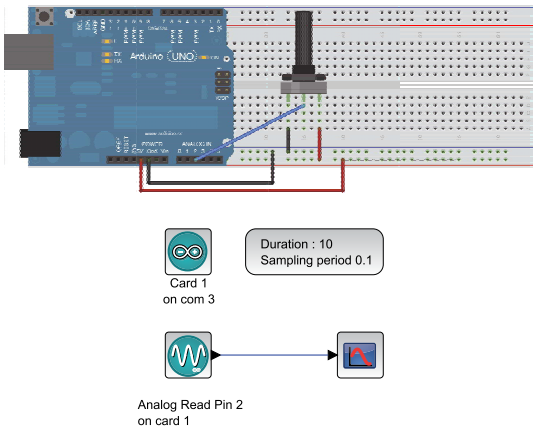
Acquérir une grandeur analogique

Dans cette partie, nous allons utiliser un potentiomètre afin de générer une tension variant entre 0V et 5V (principe du pont diviseur de tension). On pourrait acquérir la tension de n'importe quel capteur analogique, mais il faut faire attention à ne pas l'alimenter à une tension supérieure à la tension de référence (5V pour le modèle UNO ou 3.3V suivant les modèles) supportée par les PINS de l'Arduino.
L'acquisition analogique se fait grâce à un CAN (Convertisseur Analogique Numérique) de 10 bits, c'est-à-dire que la plage 0V – 5V est convertie en 2^{10} = 1024 nombres numériques, soit une plage allant de 0 à 1023. La résolution est donc de 4.9 mV.

Branchez le potentiomètre comme suit :
Patte extérieure à 0V (masse ou PIN GND), patte intérieure à 5 V et patte du milieu à relier à une entrée analogique quelconque notée Apin.

Créez un diagramme avec un port analogique en lecture Analog_READ_SB (sous-palette Analog) et double-cliquez sur l'entrée analogique pour spécifier votre numéro de PIN.
Ajoutez un afficheur ARDUINO_SCOPE (sous-palette Configuration).

Lancez l'acquisition via le bouton de simulation et faites tourner le potentiomètre.
On doit voir une courbe variant de 0 à 1023 selon l'angle du potentiomètre.



Démonstrations / Arduino / Exemples Livret / Exemple 1

Maintenant que l'acquisition d'une grandeur analogique a été validée, nous allons commander les sorties analogiques de l'Arduino.

Commande de sorties analogiques : PWM

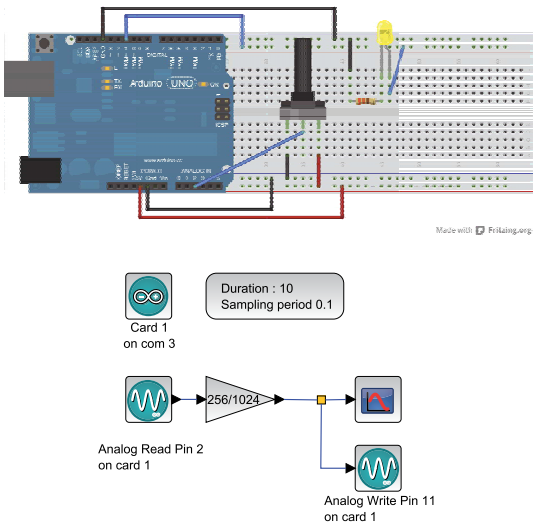
Les sorties analogiques de la carte Arduino Uno sont disponibles sur les ports des sorties logiques 3, 5, 6, 9, 10 et 11. Parler de sorties analogiques est donc un peu un abus de langage. En effet, pour générer cette sortie en minimisant les pertes d'énergie, l'Arduino utilise des PWM.

Le but est de réaliser un variateur de lumière à partir de la consigne issue du potentiomètre grâce à la diode.

Comme il s'agit d'un système très rapide (le temps de réponse est de l'ordre de la nanoseconde), elle va donc clignoter à la fréquence de 490 Hz avec un rapport cyclique variable. Ce sont nos yeux qui vont jouer le rôle de filtre en ne retenant que la valeur moyenne de l'intensité lumineuse perçue.

Ajoutez au diagramme Xcos précédent un bloc sortie analogique ANALOG_WRITE_SB (sous-palette Analog). Double-cliquez pour spécifier que cette sortie sera associée au PIN PWM de votre choix et branchez la diode sur ce PIN et sur le PIN Gnd (masse).

L'entrée du bloc sortie analogique correspond à la valeur du rapport cyclique qui peut varier de 0 (0 %) à 255 (100 %). Il est possible de mettre une valeur supérieure à 255, mais cela n'aurait pas de sens car le rapport cyclique restera à son maximum (100 %).



Démonstrations / Arduino / Exemples Livret / Exemple 2

Nous avons vu précédemment que la consigne issue du potentiomètre varie de 0 à 1023. Il faut donc adapter cette consigne pour profiter au maximum de la plage de réglage allant de 0 à 255.

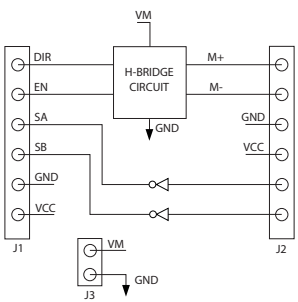
Ajoutez un gain GAINBLK_f de 256/1024 disponible dans la palette standard (sous-palette Opérations mathématiques) puis reliez l'entrée potentiomètre à la sortie PWM en passant par le gain et enfin lancez l'acquisition via le bouton de simulation afin de suivre l'évolution de la structure ainsi créée.

On constate que l'on peut maintenant commander l'intensité lumineuse de la diode en agissant sur le potentiomètre.

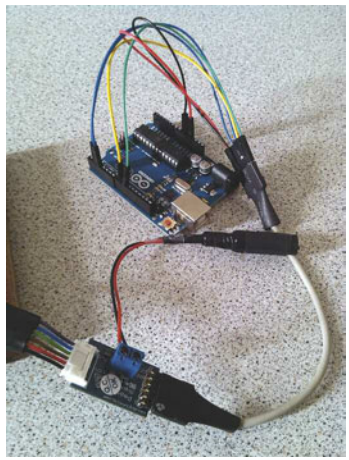
Commande en boucle ouverte d'un moteur à courant continu

Pour réaliser la commande d'un moteur, nous allons utiliser un hacheur externe. Cela est nécessaire car la carte Arduino Uno utilisée ne peut délivrer suffisamment de puissance pour alimenter un moteur.
Le hacheur fonctionne selon le même principe que le signal PWM précédent. Il hache la tension issue de l'alimentation externe grâce à des transistors. La tension moyenne dépend alors du rapport cyclique. C'est justement le signal PWM de l'Arduino qui est utilisé pour commander ces transistors.
Il est possible d'utiliser des cartes toutes prêtes qui permettent de relier simplement l'Arduino, le hacheur externe et le moteur. Il n'est cependant pas beaucoup plus compliqué de créer son propre hacheur en utilisant des circuits imprimés comme le L293D ou SN754410.
Nous allons utiliser, dans un premier temps, le moteur-réducteur 6 V ainsi que la carte PMODHB5 disponible sur plusieurs sites d'achat en ligne.
Il y a 6 broches à connecter via la rallonge sur la carte Arduino, il s'agit du connecteur J1 du schéma suivant. La rallonge possède un détrompeur (petite étoile pour le PIN 1) pour bien repérer les différentes connectiques du port J1.

Schéma de câblage de la carte PMODHB5 :
Attention, une inversion des bornes pourrait endommager le circuit.

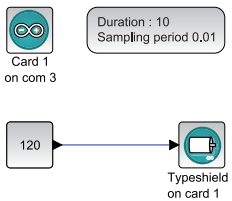


- Le signal PWM pour la commutation du pont en H sera envoyé sur l'entrée EN.
- Spécifiez le sens de rotation avec l'entrée DIR.
- SA et SB correspondent aux signaux des deux capteurs en quadrature de phase.
- L'alimentation des capteurs se fait entre GND et VCC.
- VM et GND correspondent à l'alimentation externe (chaîne d'énergie). Cf. photo ci-contre.



Reliez l'alimentation 5V à la borne VCC, ainsi que la masse à la borne GND avec des straps.
Connectez la borne DIR sur une sortie digitale de l'Arduino.
Connectez la borne EN sur une sortie PWM de l'Arduino.

On va maintenant créer l'interface avec Scilab / Xcos et le module Arduino.
Ouvrez une nouvelle fenêtre d'édition Xcos et ajoutez le bloc de configuration du port série ARDUINO_SETUP (sous-palette Configuration) et le bloc échantillonnage TIME_SAMPLE (sous-palette Configuration) avec un échantillonnage de 0.01s.
Ajoutez un bloc entrée constante CONST_m disponible dans la palette standard (sous-palette Sources) et réglez-le à une consigne comprise entre -255 et 255 (ici 120). Si la consigne dépasse 255, le PWM de l'Arduino saturera à la valeur maximale (255).



Ajoutez un bloc moteur continu DCMOTOR_SB (sous-palette Motor) et configurez-le pour gérer la carte PMODHB5 et le moteur n°1.

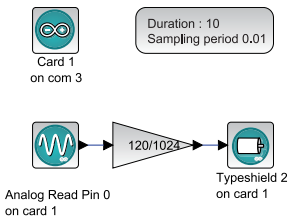
Démonstrations / Arduino / Exemples Livret / Exemple 3

Spécifiez les ports choisis (par exemple, direction : 8, commutation / PWM : 11) puis lancez l'acquisition via le bouton de simulation. Le moteur se met à tourner dans un sens.

Commande d'un moteur à courant continu avec un potentiomètre

Reliez le potentiomètre pour effectuer une acquisition sur l'une des entrées analogiques de la carte Arduino Uno.

Dans Xcos, modifiez le diagramme précédent pour lire la valeur du potentiomètre et imposer la consigne. Un gain de 120 / 1024 a été ajouté entre la consigne issue du potentiomètre et le moteur. En effet, les moteurs admettent une tension nominale maximum de 6V, mais l'alimentation du hacheur est en 12V. Il faut donc envoyer uniquement 50% environ de la tension maximale 12V, ce qui correspond à un rapport cyclique maximal de 120 pour se limiter à 6V avec une marge de sécurité (255 correspond à 12V) pour une valeur de consigne maximale de 1024.



Démonstrations / Arduino / Exemples Livret / Exemple 4

Lancez l'acquisition via le bouton de simulation.

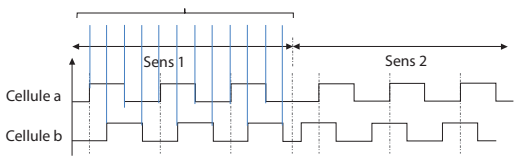
On voit que l'on commande le moteur en boucle ouverte à l'aide du potentiomètre. On se rend compte qu'il existe une forte tension de seuil avant la rotation. Cette tension est liée à d'importants frottements secs dans le réducteur.

Récupération des informations issues d'un codeur

Reprenons le schéma de câblage de la carte PMODHB5 vu précédemment. Grâce à la quadrature des deux signaux SA et SB, on arrive à déterminer le sens de rotation.

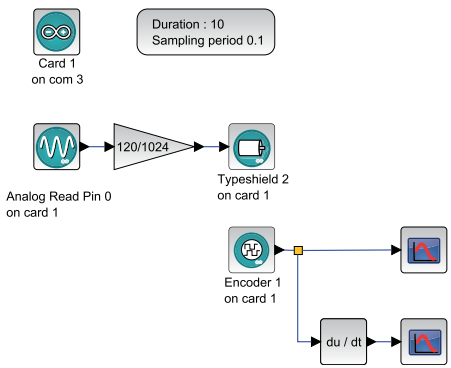
Nous allons avoir différentes options pour compter les impulsions :

- x1 : incrémente / décrémente (selon le sens de rotation) le compteur à tous les fronts montants de la voie A.
- x2 : incrémente / décrémente (selon le sens de rotation) le compteur à tous les fronts montants et descendants de la voie A.
- x4 : incrémente / décrémente (selon le sens de rotation) le compteur à tous les fronts montants et descendants de la voie A et de la voie B.



Sur les moteurs utilisés, il y a trois périodes pour chaque cellule par tour d'arbre moteur. Nous pouvons donc en x4 obtenir 12 incréments par tour de l'arbre moteur. Comme nous avons un réducteur de rapport 1 / 53, cela donne 636 incréments par tour en sortie du réducteur.
Branchez les deux broches du codeur incrémental sur les voies 2 et 3.

Pour faire fonctionner un codeur, nous avons besoin d'entrées en interruption. Ce sont des entrées capables d'arrêter le programme principal lors d'un changement d'état afin d'exécuter un sous-programme (dans notre cas incrémenter un compteur).



Démonstrations / Arduino / Exemples Livret / Exemple 5

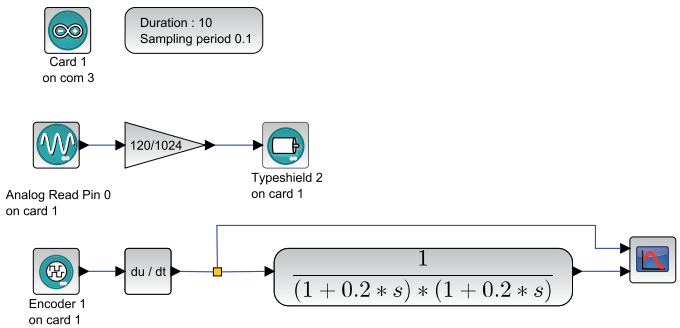
Les PINS 2 et 3 sont les seuls PINS de la carte Arduino Uno supportant les interruptions. Pour faire fonctionner un codeur en x4, il faut que les deux voies soient câblées sur les PINS d'interruption. Par contre, en x1 ou x2, seule une voie doit être branchée en interruption, l'autre peut être reliée à une entrée logique classique.

Dans Xcos, ajoutez le bloc ENCODER_SB (sous-palette Digital). Double-cliquez sur celui-ci et choisissez le numéro de la carte (1) et le mode de fonctionnement (x1, x2, x4). Ensuite, il est nécessaire de spécifier les bons PINS pour la voie A (forcément 2 ou 3) et la voie B, qui permet de trouver la direction.
Ajoutez un scope.

Positionnez un gain permettant de limiter la consigne à 120, sachant que la valeur lue par le potentiomètre varie de 0 à 1024.

Lancez l'acquisition via le bouton de simulation et testez la commande avec le potentiomètre.

Choisissez une valeur fixe de commande avec le potentiomètre et vérifiez les rapports de multiplication x1, x2, x4 en regardant la courbe obtenue.

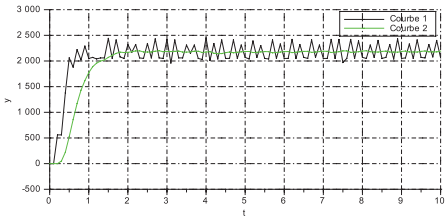


Démonstrations / Arduino / Exemples Livret / Exemple 6

À noter
Si la position est négative, il faut inverser les voies physiquement sur les PIN2 et PIN3 de l'Arduino ou inverser les PIN dans la configuration des blocs ENCODER_SB et DCMOTOR_SB.

On peut maintenant essayer d'afficher la vitesse de rotation du moteur. Pour cela, il faut dériver la position à l'aide d'un bloc DERIV (sous-palette standard Systèmes à temps continu).
On doit observer que la vitesse est assez bruitée.

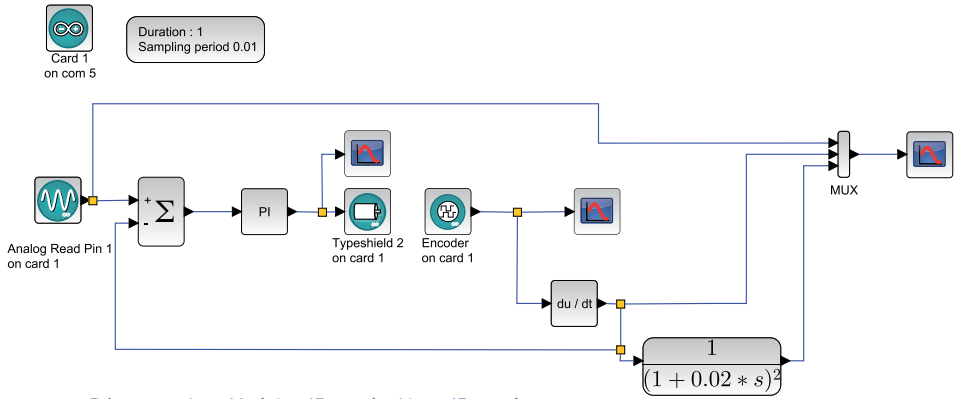
Pour filtrer le bruit, on peut utiliser un filtre du second ordre CLR, disponible dans la palette standard (sous-palette Systèmes à temps continu), comme indiqué sur la figure suivante.



La valeur de 0.2 du filtre, qui correspond à 2 fois la période de l'acquisition Arduino permet d'éviter les repliements de spectres (théorème de Shannon).

Asservissement en vitesse d'un moteur

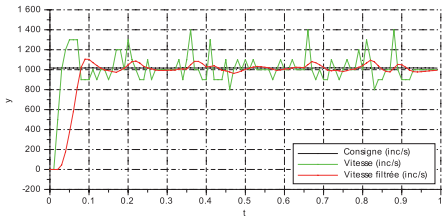
Le potentiomètre permet de définir la consigne de vitesse du moteur en incréments par seconde. Cette consigne est comparée à la vitesse réelle, obtenue par dérivation de la position fournie par le codeur incrémental, pour former un écart. L'écart est ensuite adapté à l'aide d'un correcteur de type proportionnel intégral (PI) qui fournit la consigne au moteur.



Démonstrations / Arduino / Exemples Livret / Exemple 7

On notera que pour que l'asservissement soit possible, il faut mettre une fréquence d'échantillonnage assez faible (de l'ordre d'un dizième du temps de réponse du système en boucle ouverte).

L'asservissement est également réalisé sur la vitesse brute car le filtre introduit un retard dans le calcul de la vitesse.



www.scilab.org