TNG015 Signaler och System

VT 2024

Laboration 2: Övningar med Euler/ Laplace och z-transform

**Namn: William Gabriel**

**LiU ID: wilga619**

**Epost: wilga619@student.liu.se**

**Program: Medieteknik**

**Namn: Samuel Hellqvist**

**LiU ID: samhe463**

**Epost: samhe463@student.liu.se**

**Program: Medieteknik**

## Inledande ord om laborationen

För kursens samtliga laborationer gäller att du framför allt ska satsa på att förstå frågeställningar och metoder. Ett tips är att du arbetar i din egen takt. Experimentera gärna om du får några uppslag under laborationens gång.

Den här laborationen handlar om tidskontinuerliga- och tidsdiskreta system. Du får öva på att studera egenskaper hos system (impulssvar, stegsvar) samt att forma om en differential-ekvation till en mer hanterbar differensekvation.

Redovisningen av resultat görs på anvisade platser i laborationshandledningen. Platser där grafer ska redovisas är markerade, men man kan även inkludera grafer på andra ställen vid behov.

## Förberedelseuppgifter

Läs relevanta avsnitt om Laplace och z-transformen i kompendiet eller fördjupningsboken. Dessutom ingår avsnitt som handlar om digitala filter, vilket du bör kunna åtminstone översiktligt.

## Uppgifter som ingår i laborationen

Obligatoriska uppgifter: 1 – 11

## Labbrapport inlämningsinformation

Lämna in labrapport som **pdf** fil på Lisam Submission websidan. Inkludera dina teoretiska lösningar och simuleringsresultat under varje uppgift i detta dokument och spara som pdf fil. Niu får antigen att ta foton av teoretiska lösningar på paper och inkludera dem i Word dokumentet eller skriva ekvationerna direkt med Equation Editor i Word eller använda Latex.

## Tidskontinuerliga System: Laplace-transformen

1. Ett tidskontinuerligt system kan beskrivas med differentialekvationen (systembeskrivningen):

Du vet också att systemet saknar energi initialt, dvs .

Ekvationen är tämligen komplicerad och ett sätt att beräkna utsignalen som funktion av insignalen är att lösa ekvationen. Det här kräver naturligtvis att man har goda kunskaper i lösning av differentialekvationer.

En alternativ väg att gå för att analysera vad systembeskrivningen egentligen betyder, är att gå via Laplaceverktyget.

Gör en transformering av differentialekvationen via Laplaceverktyget (Laplacetransformen) så att du får ett uttryck för överföringsfunktionen . Skriva överföringsfunktionen .

1. Överföringsfunktionen , som också är en systembeskrivning, ger dig information om det tidskontinuerliga systemet på liknande sätt som differentialekvationen gör. Men en avgörande skillnad är att det nu går att se helt andra egenskaper för systemet. Bland annat är systemets poler och nollställen mer eller mindre uppenbara.

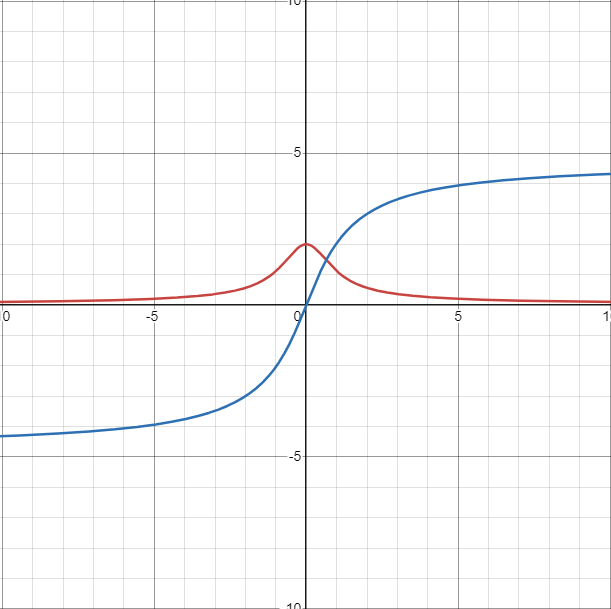
Ange systemets poler och nollställen.

Nollställen: z = -2

Poler: p1=p2=1

1. Baserat på vad du vet om pol/nollställe placeringen, skissa på hur frekvensgången och fasgången ser ut för systemet.

Frekvensgångsgrafen får du fram om du analyserar vektorlängderna för nollställen och poler i Laplaceplanet. På liknande sätt får du fram fasgången genom att analysera vektorvinklarna för nollställen och poler i Laplaceplanet.



Röd graf: frekvensgången

Blå graf: fasgången

Formel för frekvensgången:

Formel för fasgången:

1. Frekvensgång och fasgång kan hanteras i MATLAB/ Python på ett relativt enkelt sätt. MATLAB/ Python behöver veta hur systemet beskrivs i termer av täljare och nämnare för överföringsfunktionen (se uppgift 1). Täljaren och nämnaren för överföringsfunktionen ska först skrivas upp på formen:

där är heltal, högsta potensen av ”s” för täljare resp. nämnare är koefficienter (reella tal) resp. täljarterm och nämnarterm.

Observera att termen eller bara syftar på konstanter eftersom s-termen försvinner!

När täljaren och nämnaren är skrivna på ovanstående form, anger du täljarens och nämnarens koefficienter, som argument till MATLAB/ Python-funktionen bode(numerator. denominator) / Python funktionen signal**.**TransferFunction() and signal**.**bode() enligt följande:

MATLAB Code:

**bode([] ,[]);**

Python Code:

**from** scipy **import** signal

**import** matplotlib**.**pyplot **as** plt

sys **=** signal**.**TransferFunction**(*[]*,** ***[]*)**

w**,** mag**,** phase **=** signal**.**bode**(**sys**)**

plt**.**figure**()**

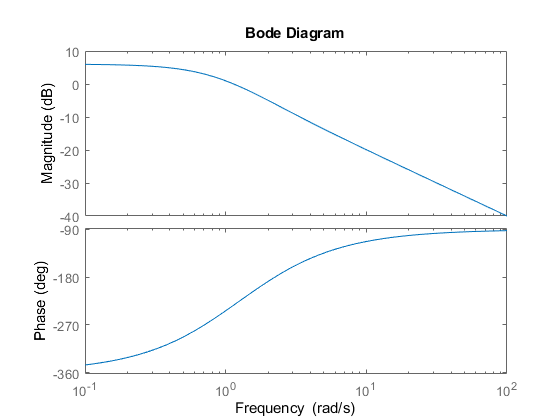
plt**.**semilogx**(**w**,** mag**)** # Bode magnitude plot

plt**.**figure**()**

plt**.**semilogx**(**w**,** phase**)** # Bode phase plot

plt**.**show**()**

Skriv in detta i MATLAB/ Python och kör funktionen bode. Det bör dyka upp två s.k. Bodediagram på datorskärmen. Den övre visar frekvensgången och den nedre visar fasgången för överföringsfunktionen . Skalorna på tre av fyra axlar är logaritmiska. Inkludera MATLAB/ Python grafer.



1. Det är viktigt att kunna avläsa diagram, som t.ex. frekvensgång och fasgång, på ett korrekt sätt. Utgå från Bodediagrammet i föregående uppgift 4. Omvandla decibel till ”linjära värden” där det behövs. Observera också att den horisontella skalan anger vinkelfrekvens (vinkelhastighet) med enheten rad/s. MATLAB/ Python anger skalan med ”frequency” men det borde rimligen stå ”angular frequency”.

Ange vilket värde de eftersökta storheterna har och fyll i tabell 1 nedan.

Tabell 1: Avläsning från Bodediagrammet för H(s).

|  |  |
| --- | --- |
| **Storhet** | **Värde** |
| rad per sek | 1.12 |
| Hz | 0.16 |
| rad per sek | 0.36 |
| *H(* då rad per sek | 4.51 |

1. Slutligen, vad gör systemet H(s), både med amplitud och fas, om insignalen exempelvis är en ljudsignal bestående av många olika toner (frekvenser)?

Du kan tänka dig en situation enligt figuren nedan.

Ljudsignal bestående av ett flertal olika frekvenser, både låga och höga

Utsignal som är ”bearbetad” av systemet H(s) på något sätt. Hur?

Skriv dina kommentarer.

Genom att titta på frekvensgångsgrafen ser vi att det är ett lågpass filter då låga frekvenser sparas och höga tas bort. Fasen är oändrad i låga frekvenser enligt fasgångsgrafen..

## Bilinjär transform: Laplace övergår till tidsdiskret z-uttryck

Utgå från differentialekvationen När du ska söka en lösning till ekvationen kan du göra det på lite olika sätt. Här följer några alternativ som är tänkbara

1. Givet en viss insignal kan du räkna fram lösningen med papper och penna om du också känner till startvillkoren (initialvillkoren). Har du lösningen kan du alltid få fram siffervärden på utsignalen.

*Fördel: en metod som går direkt på uppgiften att hitta en lösning, dvs värden.*

*Nackdel: kan bli svårt att räkna fram lösningen om differentialekv. är knepig.*

1. Skriv om differentialekvationen till en differensekvation och beräkna värden för Du måste veta insignal, initialvillkor och samplingsfrekvens.

*Fördel: Ger en enkel ekvation där y[n]-värden kan räknas fram i en tabell.*

*Nackdel: Hur väljer du bästa samplingstakt (samplingsfrekvens)?*

1. Ta fram Laplaceuttrycket för differentialekvationen och beräkna den inversa Laplace transformen. Naturligtvis bör du veta vilken insignal du har och initialvillkoren. Den inversa Laplacetransformen är alltså lösningen .

*Fördel: Omvandling differentialekv. till Laplace är ofta problemfri.*

*Nackdel: Måste därefter beräkna invers-Laplace.*

1. Ta fram Laplaceuttrycket för differentialekvationen, gör en bilinjär transformation så att du får en differensekvation. Du måste veta insignal, initialvillkor och samplingsfrekvens.

*Fördel: En lösningsväg som liknar alternativ II.*

*Nackdel: Samma som alternativ II.*

1. Använd MATLAB/ Pythons möjligheter till symboliska beräkningar och beräkna lösningen med siffervärden. Du måste veta insignal, initialvillkor och samplingsfrekvens.

*Fördel: Snabbt, relativt enkelt.*

*Nackdel: Du vet inte vad som händer i MATLAB/ Python. Finns det begränsningar du missat?*

För den enkla differentialekvation som vi har här ovanför i exemplet, skulle man kunna använda i stort sett vilken av alternativen A – E som helst. Som en övning skall du nu utföra alla fem varianterna i den här laborationen. Självklart så är vissa metoder att föredra framför andra för den här enkla ekvationen, men av pedagogiska skäl får du göra samtliga.

För uppgifterna 7 - 11 gäller att insignalen är, för och att samplingsfrekvensen sampel/sekund. Initialvillkor om .

1. Lösningar till en enkel differentialekvation Alternativ A.
   1. Beräkna lösningen till differentialekvationen manuellt genom att använda en standardformel för lösningen.
   2. Skriv ett MATLAB/ Python-program som beräknar utsignalen för ett tidsintervall sekunder och rita grafen för .
2. Lösningar till en enkel differentialekvation Alternativ B. Skriv om differentialekvationen till en differensekvation och beräkna 50 st värden på med hjälp av MATLAB/ Python. Rita resultatet
3. Lösningar till en enkel differentialekvation Alternativ C.
   1. Beräkna med papper och penna via Laplacetransformering och sedan via invers-Laplacetransformering. Utgå från den ursprungliga differentialekvationen.
   2. Rita grafen för via MATLAB/ Python-kod. Jämför graferna från denna uppgift och uppgifterna 7 och 8.
4. Lösningar till en enkel differentialekvation Alternativ D. Via den bilinjära transformationen kan man också nå fram till en differensekvation. Börja med att ta fram uttrycket för . Gör därefter en variabelsubstitution (bilinjär transform) enligt:

så att du kan skapa överföringsfunktionen .

Slutligen bör du kunna skriva uttrycket för differensekvationen genom att använda förskjutningsregeln. Testkör differensekvationen i MATLAB/ Python och rita en graf med 100 värden. Jämför med övriga grafer.

1. Lösningar till en enkel differentialekvation Alternativ E.
   1. Till sist ska du använda MATLAB/ Pythons möjligheter att räkna ”symboliskt” för att ta fram utsignalen . Du utgår i det här exemplet från Laplace-uttrycket som tidigare tagits fram.

MATLAB Code:

**>>** syms s**;**

**>>** Y**=**2**/(**s**+** 1**)^**2**;**

**>>** y**=**ilaplace**(**Y**);**

Python Code:

# import inverse\_laplace\_transform

**from** sympy**.**integrals**.**transforms **import** inverse\_laplace\_transform

**from** sympy **import** exp**,** Symbol

**from** sympy**.**abc **import** s**,** t

**from** sympy**.**plotting **import** plot

a **=** Symbol**(**'a'**,** positive **=** **True)**

# Using inverse\_laplace\_transform() method

ilt **=** inverse\_laplace\_transform**(**2**/(**s **+** 1**)\*\***2**,** s**,**t**)**

**print(**ilt**)**

* 1. För att få siffervärden från den symboliska variabeln y i MATLAB/ Python måste du konvertera symboluttrycket till ett numeriskt uttryck. Det görs med funktionen eval(y) enligt koden nedan:

MATLAB Code:

**>>** **for** n **=** 1**:**1**:**50

**>>** t **=** 0.1**\***n**;** % Skapa t-värde

**>>** y\_numerisk**(**n**)=**eval**(**y**);** % Funk. “eval” utnyttjar %t-värdet

**>>** t2**(**n**)** **=** t**;** % Spara t-värde

**>>** **end**

**>>** plot**(**t2**,**y\_numerisk**);** % Rita graf

Python Code:

# import inverse\_laplace\_transform

**from** sympy**.**integrals**.**transforms **import** inverse\_laplace\_transform

**from** sympy **import** exp**,** Symbol

**from** sympy**.**abc **import** s**,** t

**from** sympy**.**plotting **import** plot

a **=** Symbol**(**'a'**,** positive **=** **True)**

# Using inverse\_laplace\_transform() method

ilt **=** inverse\_laplace\_transform**(**2**/(**s **+** 1**)\*\***2**,** s**,**t**)**

**print(**ilt**)**

plot**(**ilt**,(**t**,**0**,**5**));**

Jämför med tidigare systemsvar (y-signal)!