CHALMERS

Programmering som undervisningsverktyg för Transformer, signaler och system

Utvecklingen av läromaterialet *TSS med DSL*

Kandidatarbete inom Data- och Informationsteknik

Filip Lindahl
Cecilia Rosvall
Peter Ngo
Jacob Jonsson
Joakim Olsson

CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Institutionen för Data- och Informationsteknik
Göteborg, Sverige, Juni 2016

Programmering som undervisningsverktyg för Transformer, signaler och system
- Utvecklingen av läromaterialet *TSS med DSL*

Filip Lindahl
Cecilia Rosvall
Peter Ngo
Jacob Jonsson
Joakim Olsson

Supervisor: Patrik Jansson, Department of Computer Science and Engineering
Examiner: Niklas Broberg, Department of Computer Science and Engineering

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
DATX02-16-05
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2016

Förord

Denna rapport behandlar kandidatarbetet “Matematikens domänspecifika språk (DSLsofMath) för andra kurser” som genomfördes på Chalmers tekniska högskola under vårterminen 2016. Vi som har gjort detta kandidatarbete är fem studenter från civilingenjörsprogrammen Datateknik och Teknisk Matematik på Chalmers.

Vi vill tacka Patrik Jansson, vår handledare, som har varit väldigt hjälpsam och skickat oss i rätt riktning när vi varit vilsna. Vi vill tacka Adam Sandberg Eriksson som har svarat på frågor vi haft och funnits tillgänglig när vi undrat saker. Vi vill tacka Bo Egardt och Ants Silberberg som tagit sig tid att diskutera sina kurser med oss. Slutligen vill vi tacka vår fantastiska testgrupp som gått igenom vårt material och gett oss återkoppling för att förbättra vår produkt.

Sammandrag

Denna rapport beskriver utvecklingen av läromaterialet “TSS med DSL” med dess tillhörande programmeringskod och lösningar. Läromaterialet riktar sig främst till studenter inom det datatekniska programmet på Chalmers tekniska högskola och har som syfte att vara ett kompletterande läromedel inom signallära där man utnyttjar studenternas kunskaper inom programmering. Materialet innehåller förklarande text och teori om bland annat komplexa tal, signaler, system och transformer.
I materialet ingår även programmeringsövningar som är skrivna med domänspecifika språk i programmeringsspråket Haskell. Rapporten beskriver vidare de undersökningar och efterforskningar som gjorts för att kunna ta fram läromaterialet, samt de tester som gjorts på den färdiga produkten.

Bakgrunden till detta projekt är att kursen inom signallära, “Transformer, Signaler och System”, på det aktuella programmet under många år har haft dåliga studieresultat på grund av studenternas ovana att hantera den typ av matematik som utgör grundstommen i kursen. Detta har visats sig inte vara något som är unikt för datastudenterna på Chalmers utan det är tvärtom ett problem inom många discipliner.

Det slutliga resultatet är ett läromaterial bestående av en hemsida med förklarande text, exempel, bilder, kryssfrågor och programmeringsövningar. Hemsidan kan ses på `http://tinyurl.com/DSLsHemsida`.

Källkod och övrigt material finns på GitHub för eventuell framtida utveckling, se `https://github.com/DSLsofMath/BScProj`.

Abstract

This report describes the development of the tutorial "TSS med DSL" with associated programming code and solutions. The tutorial is directed towards students studying Computer Science at Chalmers University of Technology and the purpose of the tutorial is to be a supplementary learning material for the course "Transforms, Signals and Systems". The tutorial contains explanations and theory concerning complex numbers, signals, systems and transforms. The tutorial also includes programming exercises using domain specific languages written in Haskell. The report further describes the surveys, research and analyses done in order to produce this tutorial and the test results from the finished product.

The background for this project is that the computer science students studying the course "Transforms, Signals and Systems" have gotten poor results for several years due to the students' unfamiliarity with the kind of mathematics which is a large part of that course. Further analysis has shown this to be a general problem not unique to Computer science students at Chalmers.

The final result is a tutorial consisting of a webpage with explanations, examples, images, quizzes and programming exercises. The webpage can be accessed at `http://tinyurl.com/DSLsHemsida`

The source code and other material is also available on GitHub for possible further future development, see `https://github.com/DSLsofMath/BScProj`.

Innehåll

Ordlista

ARCS
: En didaktisk modell för att främja motivation.

DSLs of Math
: Ett pedagogiskt projekt som utvecklar kursen DAT325, Matematikens domänspecifika språk.

Git
: Ett versionshanteringsprogram.

GitHub
: En hemsida som bygger på Git.

Haskell
: Ett funktionellt programmeringspråk..

Matematikens domänspecifika språk
: Kurs som ämnar att lära ut matematik med hjälp av funktionell programmering. Kurskod: DAT325.

Mönstermatchning
: En egenskap i ett programmeringsspråk som tillåter programmerare att matcha på strukturen som en sekvens är uppbyggd av.

Reglerteknik
: En kurs på Chalmers inom reglertekniska analys- och designmetoder. Kurskod (2016): ERE103.

TSS
: En kurs på Chalmers inom signaler och system. Kurskod: SSY080.

TSS med DSL
: Undervisningsmaterialet som beskrivs i denna rapport.

Förkortningar

DSL
: Domänspecifikt språk.

DSLs of Math
: Domain-Specific Languages of Mathematics.

EDSL
: Inbäddat DSL.

TSS
: Transformer, signaler och system.

1 Inledning

Detta kapitel beskriver bakgrunden till projektet och hur det uppstod, samt projektets mål och rapportens syfte.

1.1 Bakgrund

Det finns flera studier som visar på att det matematiska språket används på ett otydligt sätt i undervisningssammanhang och att man antar att studenterna är betydligt mer bevandrade i matematiken än de oftast är [1] [2]. Detta gör att vissa grundläggande detaljer, som då antas vara allmän kunskap, utelämnas och de studenter som inte är tillräckligt vana vid matematik får inte helhetsbilden. Det huvudsakliga problemet är alltså inte språket i sig utan att det ofta används på fel detaljnivå. Hur ska man då veta på vilken nivå man bör lägga sina förklaringar? En lösning som Sussman och Wisdom föreslår i sin studie *The role of programming in the formulation of ideas* [1] är att försöka förklara matematiska koncept för en dator, om en dator förstår konceptet är förklaringen tillräckligt detaljerad.

Ett exempel på detta problem går att finna på Chalmers tekniska högskola. Där finns det en lång trend där studenterna på utbildningen för Datateknik har haft svårigheter med kurserna *Transformer, signaler och system (TSS)* och *Reglerteknik*. Svårigheterna med dessa kurser tror examinatorerna till stor del beror på att studenterna inte är tillräckligt bekväma i det matematiska språket, se *bilaga B*.

Inspirerat av artiklar av bland andra Sussman, Wisdom och Wells påbörjades år 2014 ett pedagogiskt projekt på Chalmers, *Domain-Specific Languages of Mathematics (DSLs of Math)*, vars syfte var att minska svårigheterna som studenterna hade med kurserna TSS och Reglerteknik. Projektet har resulterat hittills i en kurs som heter *Matematikens domänspecifika språk*. Kursen gavs första gången våren 2016 och är i skrivande stund i utvärderingsfasen. I kursen får studenterna angripa matematiska problem och försöka lösa dem med ett verktyg de är mer familjära med än matematik, nämligen funktionell programmering. Språket som används i kursen är Haskell. Studenterna har tidigare varit i kontakt med Haskell och språket har dessutom en notation med fokus på tydlighet som lämpar sig väl för att lära ut matematiska begrepp [3].

Detta kandidatprojekt uppstod som en avgrening ur DSLs of Math-projektet med fokus på att utveckla material till specifika kurser snarare än en allmän matematisk grund.

1.2 Rapportens syfte

Syftet med denna rapport är att beskriva utvecklingen av läromaterialet “TSS med DSL”, som är ett kompletterande läromaterial till kursen TSS för studenter på datatekniska programmet på Chalmers, samt beskriva hur det bakomliggande problemet kan kopplas till en mer generell problematik.

1.3 Projektets mål

Syftet med projektet är att underlätta för studenter inom Datateknik att ta till sig signal- och systemteoretiska ämnen genom att utnyttja studenternas kunskaper inom programmering samt även underlätta för dem att betrakta ämnet ur en programmerares perspektiv. Tanken bakom detta är att man ska göra gapet mellan datateknik och signalteori mindre och minska problemen som nämns i bakgrundsavsnittet.

Projektet är tänkt att resultera i ett läromaterial som kan fungera som ett komplement till den kurs som ges inom signalteori på den datatekniska grundutbildningen. Detta läromaterial ska innehålla förklaringar och programmeringsövningar som är anpassade för studenter på den datatekniska utbildningen på Chalmers tekniska högskola.

2 Avgränsningar

På grund av att projektet är tidsbegränsat har projektet avgränsats till att ta fram läromaterial till endast en kurs, TSS. Läromaterialet som tagits fram är ett kompletterande material och är inte tänkt att kunna ersätta kursen eller dess material. Något material till kursen Reglerteknik har inte tagits fram, annat än indirekt i form av förkunskaper.

Projektets fokus har under utvecklingen av materialet inte legat på att använda ett matematiskt korrekt språk, matematiska bevis, avancerade matematiska begrepp eller teorier. Anledningen till detta är att materialet i första hand är tänkt att främja förståelse och väcka intresse för ämnet. Detta beslut baserades även på boken *Didaktik för ingenjörslärare* där författarna, under en diskussion om aktivt lärande, hävdar att:

> "När man lär sig något kan man få det att fastna på olika sätt. En ytligt inlärd kunskap eller ett beteende som inte tränats tillräckligt glöms snart bort. Det är med andra ord viktigt att metoderna tillämpas så att djuplärande uppnås. Textens detaljer är då inte lika intressanta som textens budskap, förståelse, sammanhang och principer. Snarare än att lära sig 'det man måste' utantill för reproduktion, sker ett personligt tillägnande relaterat till tidigare kunskap" [4, s 236].

I detta projekt är det alltså viktigt att ta fram ett interaktivt material som främjar förståelse och inte fastna i detaljer.

3 Teknisk bakgrund

Detta kapitel beskriver kortfattat de områden och programvara som behövs för att bygga upp läromaterialet.

3.1 Git och GitHub

Git är ett versionshanteringsprogram som gör det möjligt för flera personer att arbeta med samma material samtidigt utan att de skriver över varandras arbete. Så länge inte två versioner av materialet innehåller ändringar som står i konflikt med varandra, till exempel att flera personer samtidigt ändrar på samma rad, slås de ihop och bildar då en tredje version vars innehåll kommer från båda de tidigare versionerna. Alla tidigare versioner av en fil sparas och kan återställas och man kan också återställa enstaka förändringar utan att behöva återställa hela filen. Skulle två versioner stå i konflikt med varandra krävs däremot en del manuellt arbete.

För ytterligare information om Git se hemsidan `https://git-scm.com`.

GitHub är en hemsida som använder Git. GitHub gör det möjligt för utvecklare av program med öppen källkod att utveckla kod och annat material offentligt. Öppen källkod innebär att den kod som programmet är uppbyggt av är offentlig och tillgänglig för alla att läsa och kontrollera.

GitHub förenklar kommunikationen mellan användare och utvecklare genom att tillhandahålla verktyg som låter användare kommentera på enskilda versioner av källkoden, följa upp ärenden och problem samt dokumentera sitt projekt. GitHub har även verktyg för att underlätta samarbete mellan utvecklare genom att man kan klona varandras projekt och bidra med egna ändringar som huvudutvecklaren kan godkänna. Eftersom materialet lagras offentligt på GitHub så är den lättåtkomlig för andra utvecklare vilket underlättar vid framtida vidareutveckling.

För ytterligare information om GitHub se hemsidan `https://github.com`.

3.2 Funktionell programmering och Haskell

Funktionell programmering är en programmeringsparadigm där de huvudsakliga byggstenarna är funktioner. Som Hughes nämner i sin välciterade *Why Functional Programming Matters* [5], handlar funktionell programmering mycket om att göra det möjligt att skriva så modulär kod som möjligt. Anledningen är att det enklaste sättet att angripa ett problem är genom att bryta ner problemet i delproblem, lösa delproblemen, och sen sammanfoga lösningarna till en helhetslösning. Hur mycket man kan bryta ned ett problem beror på hur enkelt man sedan kan sammanfoga delproblemens lösningar igen när man är klar.

Vad som gör funktioner till grundläggande byggstenar i funktionella programmeringsspråk är att de kan skickas som parametrar till andra funktioner, så kallade högre ordningens funktioner. Detta gör att vi kan bygga funktioner som exempelvis

bara tar hand om datastrukturer och skicka funktionen som löser delproblemet som en parameter.

Partiell funktionsapplikation är en annan egenskap som underlättar för funktionella programmerare att skriva mer modulär kod. Partiell funktionsapplikation innebär att funktioner som appliceras på färre argument än de är definierade för skapar en ny funktion, se exemplet i figur 1 nedan.

```
ghci> :t (+)
(+) :: Num a => a -> a -> a

ghci> :t (+ 1)
(+ 1) :: Num a => a -> a

ghci> let plusEtt = (+1)

ghci> plusEtt 2
3
```

Figur 1: Figuren visar hur Haskell hanterar partiellt applicerade funktioner. I bilden är `(+)` addition som är definierad på två numeriska värden och `plusEtt`, `(+ 1)`, är en funktion som givet ett annat numeriskt värde adderar 1 till det.

Haskell är ett funktionellt programmeringsspråk med stöd för partiell funktionsapplikation, högre ordningens funktioner, algebraiska datatyper samt lat evaluering. Lat evaluering betyder att Haskell inte evaluerar ett uttryck förrän det behövs och detta gör det möjligt för Haskell att hantera oändliga datastrukturer utan att processen kräver för mycket virtuellt minne och terminerar.

Algebraiska datatyper är typer som består av flera andra typer. Tack vare Haskells *mönstermatchning* är det lätt att definiera rekursiva funktioner som matchar på kontruerare i de algebraiska datatyperna vilket visas i exemplet i figur 2.

```
data Tree a = Leaf | Node a (Tree a) (Tree a)

mapTree :: (a -> b) -> Tree a -> Tree b
mapTree _ Leaf = Leaf
mapTree f (Node a l r) = Node (f a) (mapTree f l) (mapTree f r)
```

Figur 2: Figuren visar den algebraiska datatypen `Tree a` som beskriver en trädstruktur och den generella funktionen `mapTree` som använder mönstermatchning för att manipulera trädstrukturen.

I Haskell är det enkelt att definiera egna datatyper vilket gör att man exempelvis kan inkapsla redan existerande typer för att göra deras syfte mer tydligt. Man kan också med algebraiska datatyper implementera typer som kan fungera som ett abstrakt syntaxträd. Med högre ordningens funktioner kan man dessutom skapa generella funktioner som arbetar på dessa typers strukturer. Detta gör att Haskell lämpar sig bra som värdspråk för inbäddade domänspecifika språk.

3.3 Domänspecifika språk

Ett domänspecifikt språk (DSL) är ett programmeringsspråk som till skillnad från generella programmeringsspråk är anpassat för en specifik domän. Att skapa domänspecifika språk är ingenting nytt, det finns tvärtom många DSL som används i vid utsträckning. Exempel på några välkända DSL är HTML, MATLAB och Game Maker Language.

Det som är grundtanken med ett DSL är att det är relativt enkelt att formulera lösningar för problem inom deras domäner, exempelvis är det enkelt att strukturera upp text i HTML, utföra matrisoperationer i MATLAB eller skapa datorspel med Game Maker Language. Däremot är DSL ineffektiva inom andra områden än de som de är implementerade för. Det är till exempel inte effektivt att försöka skapa datorspel i MATLAB.

När man designar domänspecifika språk kan man välja att skapa ett fristående språk som kompileras till någon typ av maskinkod eller att utveckla sitt domänspecifika språk i ett redan existerande generellt programmeringsspråk, där beståndsdelarna i språket är funktionsanrop och konstruerare i värdspråket. Ett DSL som är implementerat på detta sätt kallas för "inbäddat DSL (EDSL)".

Det finns även olika nivåer av EDSL; så kallade djupa EDSL (deep embedding) och ytliga EDSL (shallow embedding). I introduktionen till sin artikel *Combining deep and shallow embedding for EDSL* skriver Svenningson och Axelsson [6] att man i ett djupt EDSL använder sig av algebraiska datatyper och att man med dessa bygger upp ett abstrakt syntaxträd som sedan tolkas av olika funktioner. Man definierar alltså egna typer och funktioner som bygger upp strukturer med dessa typer och skiljer på så sätt det domänspecifika språket från värdspråket. I ett ytligt EDSL försöker man däremot uttrycka sitt domänspecifika språk i termer av värdspråkets funktioner och typer. Ett uttryck i ett ytligt EDSL motsvaras alltså av ett uttryck i värdspråket och behöver inte tolkas ytterligare. Båda sätten att implementera ett EDSL har sina för- och nackdelar då det exempelvis i ett djupt EDSL är förhållandevis lätt att ge fler tolkningar av det abstrakta syntaxträdet och därmed kunna bygga vidare på det domänspecifika språket, men svårare att lägga till konstruerare eller på andra sätt ändra i typerna som bygger upp syntaxträdet eftersom man då måste ändra i de funktioner som tolkar dem. I ett ytligt EDSL är det lätt att lägga till konstruerare och typer eftersom man bara måste kunna representera dem i värdspråket och alltså inte behöver hantera ett syntaxträd, men å andra sidan är dess tolkning fixerad. För att lägga till nya sätt att tolka ett ytligt EDSL måste det implementeras om på nytt.

3.4 Transformer, signaler och system

Inom ämnet signalbehandling beskriver kursen TSS olika typer av signaler, hur signalerna relateras till ett system och hur man kan modellera denna relation mellan signaler och systemet i matematiska termer. Ett system kan till exempel vara en fjäder vars insignal är en kraft som påverkar fjädern utifrån.

I kursen, och så även i detta projekt, begränsas signalen till enbart en oberoende variabel som beskriver tid och de system som betraktas är linjära och tidsinvarianta, så kallade LTI-system. För att kunna lösa problem med denna typ av signaler och system så kan man utveckla signalen i en Fourierserie eller använda sig av olika typer av transformer. En av de transformer som tas upp i kursen är Fouriertransform som gör det möjligt att byta perspektiv och betrakta signalen i frekvensdomän istället för tidsdomän.

Fouriertransform i kontinuerlig tid är definierad enligt följande:

$$\mathcal{F}[\ f\ ]\ \omega = \int_{-\infty}^{\infty} f(t)e^{-j\omega t}dt$$

Området transformer, signaler och system innehåller även andra begrepp och operatorer som komplexa tal, faltning, sampling med flera. Alla dessa grundkunskaper behövs för att modellera system och signaler.

3.5 Didaktik

Didaktik är vetenskapen om undervisning. I *Didaktik för ingenjörslärare* [4] hävdar författarna att didaktik kan ses som ett komplement till pedagogik. De didaktiska frågorna handlar, enligt författarna, inte om hur studenten lär sig utan snarare om hur läraren skapar en situation lämplig för lärande.

Som nämndes i inledningen har detta projekt som syfte att utveckla ett läromaterial som kan fungera som ett komplement till kursen TSS och väcka intresse för ämnet. Därför är didaktik en viktig del av projektet.

En stor del av didaktik handlar om hur man kan påverka studentens motivation. I *Didaktik för ingenjörslärare* sammanfattas motivationens betydelse enligt följande: "Varje kurs behöver motiveras för studenterna. Med motiverade studenter blir resultatet bättre.". Boken hävdar även att långt ifrån alla studenter som går en kurs utgår från att den är läsvärd. Även i *Motivational Design for Learning and Performance* påpekas det orimliga i att anta att alla studenter är motiverade att lära sig ett ämne redan när de först kommer i kontakt med det [7]. Det är alltså viktigt att ett läromedel är uppbyggt så att det motiverar studenterna att lära sig.

Enligt boken behöver motivationsaktiviteter stödja inlärningsmålen för att vara effektiva. Motivationsmedel kan vara roliga och underhållande men om de inte engagerar studenten i läromålen och läromaterialet så hjälper de inte studenten att lära sig. Roliga aktiviteter och dylikt kan användas som ett belöningssystem men främjar i sig inte lärande. Många studenter kommer även motsätta sig eller rentav avsky aktiviteter tänkta att motivera dem om aktiviteterna inte är knytna till läromålen, detta gäller särskilt för vuxna studenter. Motivationsdesign har alltså utmaningen att göra materialet tilltalande utan att det blir ren underhållning.

Ett angreppssätt inom didaktik är *ARCS-modellen*, som beskrivs i detalj i boken *Motivational Design for Learning and Performance* [7]. ARCS är en akronym för

Attention, *Relevance*, *Confidence* och *Satisfaction*, vilket är grundpelarna i *ARCS-modellen*. Nedan följer en summering av modellens fyra grundpelare utifrån boken.

Den första grundpelaren *Attention*, uppmärksamhet på svenska, syftar till att väcka studentens uppmärksamhet och nyfikenhet. Detta åstadkoms med perceptuell stimulans, frågor och variation. Perceptuell stimulans innebär att man på något sätt ändrar i undervisningsmiljön för att väcka studentens uppmärksamhet. Dessa miljöförändringar kan vara av varierande typ som till exempel en förändring i lärarens röstläge, temperatur i en undervisningssal eller ett skämt. Denna typ av miljöförändringar väcker dock endast en tillfällig uppmärksamhet och måste därför följas av frågor och variation för att ge en varaktig effekt. Därför är det viktigt att i detta skede omedelbart följa upp med frågor som väcker studentens nyfikenhet. För att behålla studentens nyfikenhet och intresse är det även viktigt med variation i undervisningen. Oavsett hur bra struktur man har i sin undervisning kommer studenterna att tappa intresset om man inte varierar upplägget.

Den andra grundpelaren *Relevance*, relevans på svenska, syftar till att övertyga studenten om att inlärningen är personligt relevant. För att en student ska vara intresserad av att lära sig måste hen känna att instruktionen är relaterad till personliga mål eller motiv. Detta uppnås genom målorientering och anknytning till bekant kunskap. Studenter är mycket mer motiverade att lära sig saker om dessa kan hjälpa dem uppnå ett mål i nutid eller framtid, som att klara ett prov, få en befordran, eller liknande. Därför är det viktigt att studenterna vet hur det de lär sig är relaterat till deras mål. Ämnet kommer även att kännas mer relevant för studenten om det kan knytas till hens tidigare kunskap eller intresse. Även om studenter kan bli nyfikna på helt nya saker är de som regel mest intresserade av saker som på något sätt är knutna till deras tidigare erfarenheter. Konkreta exempel från familjära områden gör det hela mer relevant för studenten, detta gäller i synnerhet när man lär ut abstrakt material.

Den tredje grundpelaren *Confidence*, självförtroende på svenska, syftar både till studenters tro på att de kan ämnet och deras tro på att de kan lära sig det. Även om en student är nyfiken på ett ämne och känner att det är relevant är det möjligt att hen ändå inte är tillräckligt motiverade på grund av för mycket eller för lite självförtroende. Vissa studenter kan rentav vara rädda för ämnet. På grund av detta är det viktigt att utveckla materialet så att studenten övertygas om att hen kan lära sig ämnet. Därför behöver man med hjälp av relevanta uppgifter ge studenter en upplevelse av framgång så fort som möjligt. Detta kan vara en viktig stimulans för fortsatt motivation, förutsatt att uppgiften kräver så pass mycket ansträngning att det betyder något att lyckas med den, men inte kräver så mycket att det uppgiften leder till allvarlig oro eller hot om misslyckande.

Studentens självförtroende kan byggas upp genom lärokrav, möjlighet till framgång och personlig kontroll. Det är viktigt att sätta upp tydliga lärokrav eftersom det är en stor källa till oro och osäkerhet för studenten att inte veta vad som förväntas av hen. Efter att ha skapat en förväntan av framgång är det viktigt att ge studenten möjlighet att lyckas med krävande och meningsfulla uppgifter. Dessa uppgifter bör se olika ut beroende på vilken nivå studenten befinner sig på. Studenter som är nya inom ett område reagerar generellt bäst på om att ha en relativt låg svårighetsnivå

med frekvent återkoppling som hjälper dem lyckas eller bekräftar deras framgång. När studenter bemästrat grunderna är de redo för mer krävande uppgifter. Slutligen är det viktigt att studenten känner att hen har personlig kontroll över lärandesituationen. Att uppleva att man själv kan kontrollera utgången av en situation är avgörande för självförtroendet. För att förstärka motivationen bör instruktören förse studenten med en stabil lärandemiljö och sedan låta studenten ha personlig kontroll över sin inlärning. Det är viktigt att skapa en miljö där det är acceptabelt och helt i sin ordning att göra misstag och lära sig av dem.

Om man använder de första tre grundpelarna, Attention, Relevance och Confidence, väl, kommer studenten vara motiverad att lära sig. Det är för att upprätthålla denna motivation som den sista grundpelaren *Satisfaction*, tillfredsställelse på svenska, används. För att studenten ska ha ett fortsatt intresse av att lära måste hen känna tillfredsställelse med inlärningens process eller resultat. Denna tillfredsställelse kan uppnås genom naturliga konsekvenser och allmänna positiva konsekvenser. Med naturliga konsekvenser menas den kunskap som blir en direkt följd av undervisningen, att kunna lösa en uppgift eller dylikt som man inte kunde lösa innan ger en tillfredsställelse i sig. Det är ett starkt belöningsverktyg att låta studenten använda ny kunskap. Allmänna positiva konsekvenser kan vara materiella belöningar, som till exempel en löneförhöjning, eller symboliska belöningar som diplom eller ett bara ett erkännande av studentens förmåga.

Hur dessa didaktiska modeller och metoder har använts i läromaterialet TSS med DSL beskrivs i avsnitt *6 - Utveckling av läromaterialet*.

3.6 Relaterad forskning

Detta avsnitt beskriver i korthet tre studier som är nära relaterade till detta projekt.

En studie som studerat undervisning med det matematiska språket är *The role of programming in the formulation of ideas* av Gerald Sussman och Jack Wisdom. Sussman och Wisdom uttrycker att det är det svårt att lära sig fysik: de resonerar att detta troligen kan bero på att studenter tycker att det är svårt med det matematiska språket och att det därför blir problematiskt att lära sig både ämnet och språket som ämnet är uttryckt i [1]. De jämför det med att försöka läsa *Les Misérables* samtidigt som man försöker lära sig franska.

Det huvudsakliga problemet med att uttrycka idéer och koncept i det matematiska språket är dock inte språket i sig utan snarare att man, som med naturliga språk, antar att den man talar med eller förklarar för är van vid det matematiska språket och därför delar samma kunskapsbas om matematik som man själv. Detta gör, enligt Sussman och Wisdom, att man istället för att förklara matematiken så tydligt som man bör snarare skissar upp en lösning, vilken för den matematiskt ovane studenten blir i det närmaste obegriplig.

I artikeln föreslås en lösning på hur man hittar den detaljnivå som krävs för att otvetydigt förklara ett matematiskt koncept, nämligen att försöka förklara det för en dator. En dator försöker inte tänka själv och accepterar inte halvfärdiga instruktioner där resten lämnas till intuitionen.

En annan studie som är intressant för projektet är *Communicating mathematics: Useful ideas from computer science* av Charles Wells [2]. I denna artikel argumenterar Wells för att matematiker har mycket att vinna på att uttrycka sig tydligare. Wells drar paralleller till datavetenskapen där man skiljer noga på syntax och semantik samt tänker explicit på typer och hur dessa manipuleras. Detta skiljer sig från matematiken där man inte separerar på semantik och syntax i samma grad och där typerna oftast är implicita och lämnas till läsaren att själv fundera på.

Detta kandidatprojekt bygger, som tidigare nämnts, på projektet DSLs of Math som i sin tur bygger på idéerna från Wells, Sussman och Wisdom. I kursen som utvecklats i projektet, *Matematikens domänspecifika språk*, får studenterna angripa matematiska problem ur ett programmeringsperspektiv genom att bygga domänspecifika språk. För att kunna göra detta måste studenterna fundera över vad de faktiskt ska modellera och vilka typer de kan tänkas behöva för att uttrycka problemet med [8].

4 Problemanalys

Som nämndes i *1.1 - Bakgrund* är det bakomliggande problemet till detta projekt att datastudenterna på Chalmers har svårt för kurserna TSS och Reglerteknik.

Statistiken från Chalmers visar tydligt att kurserna TSS och Reglerteknik har en ovanligt hög andel studenter som blir underkända på tentamen. Under perioden 2010 till 2016 blev i genomsnitt endast 51% av de som skrev tentamen i TSS godkända och under samma period blev 53% godkända i Reglerteknik [9]. Det finns även ett mörkertal i dessa siffror eftersom inte alla studenter som läser kursen skriver tentamen, exempelvis valde 48 av 122 registrerade studenter att inte tentera Reglerteknik [10].

Kurserna i fråga, TSS och Reglerteknik, läses under tredje året på Datateknik och kräver bland annat förkunskaper från de kurser i matematik studenterna läser under sitt första år. Reglerteknik kräver även förkunskaper från TSS. Det kan därför anses vara ett rimligt antagande att bristande kunskaper i TSS orsakar följdproblem och utgör en del av studenternas problem med Reglerteknik. Ett läromaterial som ger ökad förståelse av TSS skulle därför även kunna bidra till att ge ökad förståelse av Reglerteknik.

Kursernas examinatorer tror att studenternas svårigheter med såväl TSS som Reglerteknik till stor del beror på att studenterna inte är tillräckligt bekväma i det matematiska språket. Se *bilaga B* för en sammanfattning av hela intervjun. Examinatorerna tror att studenterna på Datateknik inte har tillräcklig vana att tolka innebörden av de matematiska beskrivningarna och är allmänt ovana vid matematiskt hantverk. De tror att datastudenterna därför lägger det mesta av sin energi på att få ihop matematiken rent praktiskt, istället för att se vad matematiken står för.

5 Produktspecifikation

För att åtgärda det problem som beskrivs i föregående avsnitt har gruppen skapat en unik produkt som är anpassad speciellt för TSS på programmet för Datateknik på Chalmers. Den ursprungliga produktspecifikationen som tilldelades vid arbetets start var relativt vag. Gruppens uppgift var att "ta fram 'DSLs of Math-inspirerat' kompletterande material för andra närliggande kurser". Det önskade implementeringsspråket var Haskell och det utvecklade materialet skulle läggas upp på GitHub.

Gruppen beslutade snabbt att fokusera på kurserna TSS och Reglerteknik av de anledningar som nämns i problembeskrivningen. Dock konstaterades det en bit in i projektet att tiden som fanns endast räckte till att utveckla material av önskvärd kvalitet till en kurs. Gruppen valde då kursen TSS, med motiveringen att TSS innehåller förkunskaper som är nödvändiga för Reglerteknik.

5.1 Produktens struktur

För att göra läromaterialet så lättillgängligt som möjligt beslutades det att materialet skulle samlas på en hemsida. På detta sätt kunde man enkelt samla text, bilder och programmeringskod och presentera dem på ett överskådligt sätt för studenterna i målgruppen.

Det beslöts att läromaterialet skulle innehålla ett antal delavsnitt som tillsammans täckte in innehållet i kursen TSS i stora drag. Avsikten med indelningen var att göra materialet mer överskådligt och även göra det möjligt för studenterna att enkelt hitta aktuellt material för ett specifikt område.

Varje delavsnitt innehåller förklaringar och exempel i löpande text. Texten ska vara skriven på ett sådant sätt att den är lätt att ta till sig för målgruppen och uppmuntrar till vidare studier av ämnet.

Delavsnitten innehåller även exempel på hur det aktuella ämnet kan beskrivas med funktionell programmering och DSL, samt programmeringsövningar.

5.2 Programmeringskod och övningar

Programmeringskoden och övningarna ska vara skrivna på ett sådant sätt att det endast krävs grundläggande förkunskaper i funktionell programmering och Haskell för att förstå dem. Studenten ska inte behöva ha läst kursen Matematikens domänspecifika språk eller motsvarande för att förstå övningarna, även om det självklart underlättar.

Koden ska vara skriven för att vara lättläst och pedagogisk snarare än att vara effektiv, eftersom koden endast är tänkt som ett läromedel och inte ska användas för att till exempel beräkna signalers Fouriertransform i andra sammanhang än i undervisningssyfte.

6 Utveckling av läromaterialet

I detta avsnitt beskrivs hur projektgruppen gått till väga under utvecklingen av läromaterialet, samt vilka verktyg och metoder som använts och hur dessa tillämpats.

Enligt boken *Motivational design for learning and performance* kan man använda sig av en modell med 10 steg vid framtagningen av nytt material till en kurs [7]. Dessa 10 steg är som följer:

1. Samla kursinformation. Hur ser situationen ut nu? Vad är nuvarande kursbeskrivning, kursmål, etc?

2. Samla information om studenterna. Vad har studenterna för relevanta egenskaper i form av intressen och tidigare erfarenheter och förkunskaper?

3. Analys av studenterna. Vad har studenterna för motivation och syn på kursen?

4. Analys av existerande material. Finns det någon motivationstaktik i det nuvarande materialet och är den i så fall lämplig?

5. Lista mål och utvärderingar. Vad vill jag åstadkomma och hur bedömer jag om jag lyckas?

6. Lista möjliga taktiker. Vilka möjliga taktiker kan uppnå de satta målen?

7. Välj och designa taktik. Vilken/vilka taktiker passar bäst för den här publiken, ämnet etc.

8. Integrera med instruktion. Hur kan jag kombinera motivationskomponenterna med ämnesinformationen?

9. Välj och utveckla material. Hur hittar eller skapar jag material för dessa mål?

10. Utvärdera och iterera vid behov. Hur kan jag utvärdera de väntade och oväntade effekterna av resultatet?

Dessa steg har legat till grund för mycket av projektgruppens metodik vid utvecklingen av läromaterialet. Gruppen inledde i enlighet med de 10 stegen med att samla in och analysera information om kursen och målgruppen, se avsnitt *6.1 - Inledande efterforskning och förundersökning*. Därefter arbetades en taktik enligt ARCS-modellen fram, se avsnitt *6.3 - Didaktik i läromaterialet*. Gruppen utvecklade och testade materialet, se avsnitt *6.6 - Test av läromaterialet*, samt utvärderade och kompletterade det.

Gruppen hade även två inspirationskällor i form av *BetterExplained* [11] och *Learn You a Haskell for Great Good* [12]. Detta är två läromaterial som projektgruppen upplevt är både informativa och underhållande läsning.

6.1 Inledande efterforskning och förundersökning

Projektgruppen inledde med litteraturstudier och efterforskningar inom ämnet. Projektdeltagarna läste in sig på signallära, DSL, didaktik, funktionell programmering och annan relaterad forskning.

Under projektets tidiga fas intervjuades två personer ur personalstyrkan på Chalmers som undervisat och varit examinatorer i kurserna Transformer, signaler och system och Reglerteknik. Se *bilaga B* för en sammanfattning av denna intervju. Under intervjun redogjorde examinatorerna för hur de upplevde studenternas problem med kurserna och vad de trodde var orsaken.

Projektgruppen sammanställde sedan en enkät, riktad till de studenter som läst kurserna Transformer, signaler och system och Reglerteknik, eller skulle läsa kurserna inom ett år. Denna enkät arbetades fram utifrån böckerna *Enkäten i praktiken* [13] och *Enkätboken* [14].

När enkäten var framtagen användes hemsidan *webbenkater.com* för att enkelt kunna hantera och analysera studenternas svar anonymt. Länken till enkäten skickades ut till alla datastudenter från årgångarna 2012-2014 och 77 svar erhölls totalt. Se *bilaga A* för ett utdrag.

Tillslut sammanställdes den insamlade informationen om kursen, såsom kursplan, lärandemål samt de senaste årens tentor. Utifrån kursinformationen, samt intervjun och resultatet från enkätundersökningen, beslutades vilka områden som läromaterialet skulle täcka.

6.2 Läromaterialets struktur

Läromaterialet delades upp i fyra ämnesavsnitt samt en introduktion. I introduktionen beskrivs läromaterialet övergripande samt vilka förkunskaper som krävs. Där förklaras även konceptet DSL. Efter introduktionen följer det första avsnittet, vilket kort beskriver komplexa tal och Eulers formel samt introducerar studenten till det DSL som används i läromaterialet. Det andra avsnittet beskriver olika typer av signaler och deras egenskaper, det tredje beskriver LTI-system och det fjärde och sista beskriver Fourierserier och Fouriertransform. De ämnen avsnitten beskriver valdes, som nämns i föregående avsnitt, utifrån information från kursen, intervjun med examinatorerna samt studenternas enkätsvar.

Läromaterialets första ämnesavsnitt tillägnades komplexa tal, eftersom förståelse av komplexa tal är en viktig förkunskap till TSS. Detta var en anledning till att gruppen valde att inleda med repetition av detta ämne. En annan anledning till att inleda med komplexa tal var att detta är ett område datastudenter känner relativt väl till vid den tidpunkt i studierna då de läser TSS. Gruppen ville ge studenterna möjlighet att bekanta sig med det DSL som används och hur man kan implementera matematik med detta innan man började beskriva nytt material. Därför ansåg gruppen att det var lämpligt att inleda med ett ämne som datastudenterna kände sig relativt bekväma med för att de skulle kunna fokusera på programmeringsaspekten och förstå upplägget med DSL.

Det andra avsnittet tillägnades signaler. Signaler är ett viktigt begrepp i kursen TSS och kunskap om signaler är en nödvändig grund för att förstå de följande avsnitten. Därför var det ett självklart val för projektgruppen att låta andra avsnittet handla om detta. I avsnittet förklaras, något förenklat, med text och figurer vad signaler är och hur några vanliga typer av signaler, som till exempel enhetssteg, ser ut och beter sig.

Det tredje avsnittet tillägnades LTI-system. I princip alla system som beskrivs i TSS är LTI-system. Kunskap om LTI-system och dess egenskaper är därför avgörande för att förstå och räkna på de system som tas upp i TSS. Avsnittet tar även upp faltning och hur detta fungerar i LTI-system.

Det fjärde och sista avsnittet tillägnades Fourierserier och Fouriertransform. Dessa är centrala begrepp i kursen och något som många datastudenter upplever som svårt. Därför valde gruppen, då det konstaterades att det inte fanns tid att beskriva alla transformer som finns med i TSS, att fokusera på Fouriertransform. En anledning till detta var att Fouriertransform är nära besläktad med såväl Laplacetransform som Z-transform och gruppen bedömde att förståelse för Fouriertransform sannolikt skulle bidra till förståelse för de andra två transformerna.

6.3 Didaktik i läromaterialet

Läromaterialet ska vara uppbyggt så att studenten själv får arbeta med det utifrån sina tidigare kunskaper. Projektgrupen ville åstadkomma ett aktivt lärande som ger studenten långvarig förståelse och intresse för ämnet, snarare än kunskap studenten lär sig tillfälligt utantill. *Didaktik för ingenjörslärare* [4, s 236] beskriver aktivt lärande enligt följande: “Idéen bakom aktivt lärande är att en lärprocess bygger på en konkret erfarenhet som följs av reflektion och observation.” Med bakgrund av detta har projektgruppen fokuserat på att göra läromaterialet interaktivt. Därför har enklare kryssfrågor lagts in i texten med avsikten att låta studenten arbeta aktivt med ämnet när hen läser teorin. Det har även lagts in programmeringsövningar där studenten får använda och utveckla sina kunskaper i ämnet.

Det finns många didaktiska modeller som kan användas vid utveckling av läromedel, men den som valdes för läromaterialet TSS med DSL var ARCS-modellen. Eftersom läromaterialet ska fungera som ett komplement till en redan existerande kurs bedömde gruppen att det är extra viktigt att materialet är motiverande, eftersom studenten läser det vid sidan av sina vanliga studier. Därför ansåg gruppen att ARCS-modellen var ett bra val, eftersom den lägger stor vikt vid just motivation. Ytterligare en anledning till att använda ARCS-modellen var att ARCS är en didaktisk modell som var relativt lätt att anpassa för ett skriftligt material. ARCS-modellen beskrivs i mer detalj i avsnitt *3.5 - Didaktik*.

ARCS-modellens första grundpelare, *Attention*, har praktiserats i läromaterialet bland annat genom att använda skämtsamma formuleringar, sammanfattningar och exempel inom ämnet. Projektgruppen har strävat efter att hålla en lättsam och

humoristisk ton för att hålla studenternas uppmärksamhet. Dock har lustiga utvikningar, formuleringar och exempel som avviker från ämnet eller inte fyller sin funktion, att främja intresse och förståelse för ämnet, undvikits. För att sedan väcka ett mer varaktigt intresse hos studenten ställes frågor i löpande text och övningar. Varje avsnitt inleds med en fråga av typen "Vad är det här egentligen för något och vad används det till?". För att behålla studentens intresse genom hela läromaterialet har upplägget på avsnitten varierats till viss del. Eftersom det är eftertraktansvärt att behålla en läsvänlig och överskådlig struktur i hela läromaterialet har materialet främst varierats med olika typer av exempel och uppgifter. I ett kapitel får studenterna fylla i ett jämförelsevis stort antal kryssfrågor och i ett annat fokuseras istället på en viss typ av programmeringsövningar där studenten får fylla i saknad kod. Målet har alltså varit att främst göra studenternas eget arbete omväxlande för att att de inte ska bli uttråkade och ge upp.

Den andra grundpelaren, *Relevance*, har praktiserats främst i inledningen av avsnitten och i introduktionsavsnittet. I introduktionsavsnittet förklaras att läromaterialet finns till för att vara ett komplement till, och hjälpa datastudenterna med, TSS. Varje ämnesavsnitt inleds sedan med en kort beskrivning av ämnet och vad det kan användas till. På detta sätt förklaras först varför läromaterialet som helhet är relevant för studentens studier och sedan förklaras vad de olika ämnesområdena kan användas til mer i detalj efterhand som studenten stöter på dem i materialet.

Den tredje grundpelaren, *Confidence*, har praktiserats genom att skapa övningar och uppgifter av varierande svårighetsgrad och ett upplägg där studenten själv kontrollerar sin inlärning. Beträffande övningarna så inledes läromaterialet i det första avsnittet med enklare kryssfrågor och relativt enkla programmeringsövningar. Kryssfrågorna kan kryssas i direkt på hemsidan, varpå de rättas automatiskt för att ge studenten omedelbar bekräftelse. Det finns även lösningsförslag till programmeringsövningarna som studenten kan jämföra sin egen kod med alternativt få hjälp av om hen kör fast. Eftersom det är önskvärt att studenten själv ska kunna kontrollera sin inlärning och arbeta i en takt som fungerar för hen passar det bra att lägga upp läromaterialet i form av en hemsida. På så vis har studenten tillgång till all information och alla uppgifter men kan arbeta med materialet som hen vill.

Den fjärde och sista grundpelaren, *Satisfaction*, har praktiserats främst i slutet på avsnitten och i uppgifterna. I kryssfrågorna får studenten omedelbar bekräftelse när hen har lärt sig något eller kan det sedan innan. Efter att studenten har läst teorin inom ett område följs detta upp med övningar där studenten får praktisera sin nyvunna kunskap. Varje avsnitt i läromaterialet avslutas till sist med en lista på vad studenten har lärt sig i avsnittet. Studenten får alltså både ett erkännande av sin kunskap och förmåga, samt möjlighet att praktisera denna.

6.4 Implementationen av vårt DSL

Det DSL som används i läromaterialet TSS med DSL har utvecklats i Haskell av projektgruppen. Det har designats för att täcka de valda delarna av TSS som behandlas i läromaterialet.

En av dessa delar är komplexa tal, vilket är ett ämnesområde som används flitigt när man arbetar med signaler och system. Därför implementerades komplexa tal som första byggstenen i läromaterialets DSL. Därefter implementerades signaler, LTI-system och Fouriertransform.

Utvecklingen av det DSL som använts har skett genom att all kod för ett avsnitt först utvecklats fristående från det övriga lärometerialet. När koden sedan hade testats och fungerade tillfredsställande delades den upp i kodpaket.
I dessa kodpaket har vissa funktioner tagits bort för att lämnas som övningar till läsarna att implementera. Resterande kod i paketen kan användas av studenten så att hen kan köra sin egen kod direkt och testa sina lösningar utan att behöva implementera bakomliggande kod som är irrelevant för inlärningen.

Samtliga funktionerna har givits så beskrivande namn som möjligt. Projektgruppen har även försökt åstadkomma en så lättläst syntax som möjligt där alla implementerade funktioner kan läsas och förstås av någon som behärskar grundläggande Haskell. Detta var viktigt då avsikten var att åstadkomma kod som kan förstås av hela målgruppen och inte bara av studenter som är särskilt intresserade av Haskell.

För att visa hur projektgruppen har tänkt vid implementationen av sitt DSL kommer delar av implementationen av komplexa tal nu att beskrivas. Vi börjar med hur ett komplext tal representeras, i vårt fall som två flyttalsvärden.

```
data Complex = Complex Double Double
  deriving Eq
```

Ett värde för den reella delen av talet och ett för den imaginära delen. Det komplexa talet $z = 1 + 2j$ representeras i vår DSL som:

```
Complex 1 2
```

De implementerade komplexa talen ska bete sig som matematikens komplexa tal och därför var det nödvändigt att implementera diverse beräkningsoperatorer för dem. Addition kunde implementeras relativt enkelt då de reella delarna adderas med varandra och de imaginära med de imaginära enligt formeln nedan [15].

$$(a, b) + (c, d) = (a + c, b + d)$$

Vilket i Haskell implementerades så här:

```
z + w = Complex (realPart z + realPart w) (imPart z + imPart w)
```

I kodexemplen ovan används funktioner som plockar ut den reella (`realPart`) respektive den imaginära delen (`imPart`) ur ett komplext tal.

Multiplikationen var lite mer komplicerad än additionen, då den matematiska formeln för multiplikation med komplexa tal ser ut enligt formeln nedan [15].

$$(a, b) \cdot (c, d) = (ac - bd, ad + bc)$$

Detta implementerades enligt följande i vårt DSL:

```
z * w = Complex (realZ*realW - imZ*imW) (realZ*imW + realW*imZ)
  where realZ = realPart z
        realW = realPart w
        imZ = imPart z
        imW = imPart w
```

Här skapades även hjälpdefinitioner, vilka är de funktioner man finner efter **where** i koden. Detta var för att undvika att skriva samma kod om och om igen, samtidigt som det gör koden mer tydlig och lättläst för studenten.

Efter att dessa operationer var färdiga så var division nästa stora operator att implementera. Den matematiska formeln för detta ser ut som följer:

$$z/w = (z \cdot w')/(w \cdot w')$$

där w' är w:s konjugat. Man förlänger alltså både täljare och nämnare med nämnarens konjugat, vilket innebär att nämnaren blir helt reell. För att implementera division var gruppen därför tvungen att först implementera en funktion som tar fram konjugatet av ett komplext tal. Detta innebär att man byter tecknet på den imaginära delen av talet. Vår funktion för det såg ut så här:

```
conjugate z = Complex (realPart z) (negate (imPart z))
```

Vår implementation av division blev då följande:

```
z / w = Complex (realPart zw' / realPart ww')
                (imPart zw' / realPart ww')
  where zw' = z * (conjugate w)
        ww' = w * (conjugate w)
```

En viktig förutsättning för att kunna koppla de komplexa talen till kursen TSS var att man först implementerade Eulers formel, som kan användas för att skapa komplexa tal utifrån en vinkel ϕ (phi). Eulers formel ser ut så här:

$$e^{j\phi} = \cos\phi + j \cdot \sin\phi$$

Implementationen av den här representationen skedde i flera steg. Funktionen `euler` skapar ett komplext tal utifrån en given vinkel, representerad av ett flyttal.

```
euler phi = Complex (cos phi) (sin phi)
```

Enligt potenslagarna så är $e^{a+jb} = e^a \cdot e^{jb}$ och enligt Eulers formel så är $e^{jb} = \cos b + j \cdot \sin b$. Exponentialfunktionen för ett komplext tal kan därför implementeras enligt nedanstående kod:

```
exp z = scale (exp (realPart z)) (euler (imPart z))
```

Sedan implementerades även trigonometriska funktioner med hjälp av Eulers formel. $\sin(x)$ representeras enligt följande som ett komplext tal:

$$\sin(x) = (e^{jx} - e^{-jx})/2j$$

Vilket ser ut så här i vårt DSL:

```
sin z = (exp (j*z) - exp (-(j*z))) / (scale 2 j)
 where j = Complex 0 1
```

Den fullständiga koden finns på GitHub:

`https://github.com/DSLsofMath/BScProj/tree/master/Tutorial/Kod`

Ett av de största problemen vid implementationen av signaler, och all kod som bygger på signaler, var hur kontinuerlig tid skulle hanteras. Kursen TSS behandlar både diskreta och kontinuerliga signaler och det var därför önskvärt att implementera båda dessa. En dator kan dock under normala förhållanden inte hantera kontinuerliga funktioner utan endast diskreta.

Ett sätt att hantera detta är att simulera ingående data till en kontinuerlig funktion med diskret data, där man låter avstånden mellan de diskreta värden datan kan anta vara väldigt små. Ju mindre dessa steg är desto bättre approximation av en kontinuerlig funktion erhålles. Frågan som uppstår är hur lämplig en diskret approximation är för att främja förståelsen av signaler mm i kontinuerlig tid. Se avsnitt *8 - Diskussion* för vidare diskussion kring detta.

Ett annat sätt att implementera kontinuerliga fall är med djup inbäddning och sen använda mönstermatchning för att implementera en funktion för olika signaler eller annan indata. Funktioner som implementeras på detta sätt ger svar som är mer matematiskt korrekta än en diskret approximation. Detta ger dock inte samma möjlighet att se mönster och att sammanfatta området i en funktion som med en diskret approximation.

Projektgruppen valde till slut en kompromiss av dessa två angreppssätt och använde både diskreta approximationer och mönstermatchning för att beskriva kontinuerliga fall. Gruppen har växlat mellan varianterna beroende på vilken av dem som föreföll ge den tydligaste förklaringen av det aktuella ämnet. Exempelvis har approximation med diskret data använts i kapitlen om signaler och system och i kapitlet om fouriertransform har djup inbäddning och mönstermatchning använts.

6.5 Presentation av läromaterialet

Efter att utkast till läromaterialets avsnitt utvecklats gjordes dessa om till LaTeX-format. Anledningen till detta var att det var ett förhållandevis enkelt sätt att få en överskådlig struktur och ett bra utseende på läromaterialet.

När avsnitten i läromaterialet var färdiga så skrevs de om till HTML, med hjälp av Pandoc, för att kunna skapa en hemsida för att materialet lättåtkomligt. Pandoc är ett program som översätter text mellan olika språk. Endast ett fåtal manuella kompleteringar krävs. En annan fördel med att arbeta med HTML-sidor i stället för

PDF är att man kan använda sig av JavaScript, vilket ger möjligheten att kunna göra uppgifterna interaktiva. Vid ändringar på sidan så redigeras det direkt i HTML koden, detta för att behålla strukturen på hemsidan. Om projektgruppen hade haft mer efarenhet av att arbeta med HTML skulle man kunnat redigera i LaTeX för

att sedan generera HTML koden igen, men det hittades ingen simpel lösning till att göra detta och samtidigt behålla strukturen på läromaterialet.

För uppgifterna i läromaterialet användes jQuery, vilket är ett JavaScript- bibliotek, för att göra det lättare att arbeta med JavaScript. Anledningen till detta var att projektgruppen ansåg att JQuery var mer lättläsligt än vanlig JavaScript och det var därmed lättare att felsöka i koden.

Beträffande designen av hemsidan användes verktyget Bootstrap. Detta för att Bootstrap har CSS kod, som används vid design av hemsidor, som man kan använda sig av. Bootstrap gör det även lätt att skapa en responsiv hemsida, vilket gör att hemsidan skalas om efter skärmstorleken utan att spendera mycket tid för att få en läsbar design. På grund av den responsiva hemsidan blir det möjligt att använda den skärm man har tillgänglig, vare sig det är en dator eller mobil, utan att hemsidan blir dåligt optimerad. På detta vis undviks att läsaren distraheras bort från texten på grund av dålig optmering.

De figurer som gjorts till läromaterialet i form av funktionskurvor och liknande är skapade i Matlab och `fooplot.com/`.

6.6 Test av läromaterialet

För att få så givande testsvar som möjligt var det givetvis önskvärt att testa läromaterialet på den aktuella målgruppen. Det var dock inte möjligt att testa materialet på studenter som läste kursen samtidigt eftersom kursen endast ges under höstterminen. Därför kontaktades studenter som läst kursen tidigare och studenter som skulle läsa den nästa höstttermin och hade läst de kurser som är nödvändig förkunskap. Dessa studenter kontaktades i samband med enkäten som beskrivs i avsnitt *6.1 - Inledande efterforskning och förundersökning*. De studenter som sedan var villiga att delta i testandet av vår produkt lämnade en e-postadress. Detta resulterade i en testgrupp på 13 personer.

När projektgruppen ansåg att läromaterialets tre första avsnitt var redo att testas skickades dessa ut till testgruppen. Testningen fungerade så att testpersonerna fick ta del av materialet, arbeta med detta och sedan ge återkoppling till projektgruppen. Det material som testgruppen fick innefattade texter, exempel, programkod och uppgifter samt lösningar till uppgifterna.

När studenterna hade testat produkten och återkopplat med vad de tyckte var bra och vad de tyckte var dåligt så bearbetades och analyserades deras återkoppling. Utifrån testgruppens kommentarer implementerades förbättringar för att få läromaterialet mer lättläst och bättre anpassat till målgruppen.

Det sista avsnittet skickades inte ut till testgruppen utan utvärderades endast internt av gruppens medlemmar då detta avsnitt inte blev färdigt i tillräckligt god tid för att ge testgruppen en rimlig chans att arbeta med materialet. Den mer generella återkoppling som gavs på de tre första avsnitten tillämpades dock även på det sista avsnittet.

7 Resultat

Projektgruppens uppgift var, vilket beskrivs i detalj i avsnitt *5.1 - Produktens struktur*, att ta fram ett kompletterande läromaterial till TSS inspirerat av *DSLs of Math*. I detta avsnitt beskrivs det resulterande läromaterialet och resultat från testerna.

7.1 Läromaterialet

Läromaterialet består av fyra avsnitt samt en introduktion. Avsnitten innehåller förklarande text, bilder, kodexempel och uppgifter. Varje avsnitt har även ett tillhörande kodpaket med förimplementerad kod till övningarna och lösningar till dessa. De fyra avsnitten behandlar i huvudsak komplexa tal, signaler, LTI-system och Fouriertransform. Ytterligare avsnitt, som beskriver Laplacetransform och Z-transform, var planerade och påbörjade men detta fick bortprioriteras på grund av tidsbrist.

Läromaterialet finns samlat på en hemsida, där det även är möjligt att ladda ner materialet i form av en PDF-fil. Följ länken nedan för att se hela materialet: `https://cdn.rawgit.com/DSLsofMath/BScProj/master/Hemsida/index.html`

Bilaga 3 innehåller ett delavsnitt ur läromaterialet som beskriver *Udda och Jämna signaler*. De övriga avsnitten har samma grundläggande upplägg som detta. Delavsnittet inleds med övergripande definitioner och förklaringar av vad som menas med udda och jämna signaler inom signallära, detta illustreras sedan med figurer. Därefter följer kryssfrågor där studenten ska avgöra om den givna signalen är jämn, udda eller ingetdera. Detta följs av ett kodexempel där studenten får se hur man kan implementera en funktion som avgör om en signal är udda. Studenten får sedan implementera motsvarande funktion för jämna funktioner själv som en övning. Till sist ges studenten två signaler att testa sina programmeringsfunktioner på, både för detta och föregående delavsnitt.

7.2 Testresultat

Introduktionen och de tre första avsnitten i läromaterialet har granskats av den testgrupp som beskrivs i avsnitt *6.6 - Test av läromaterialet*. Det sista avsnittet har inte granskats av testgruppen, utan endast av projektgruppen själva samt projektgruppens handledare. Orsaken till detta beskrivs även det i avsnitt *6.6 - Test av läromaterialet*.

Responsen från projektets testgrupp var genomgående positiv. Testgruppen hävdade att de trodde att läromaterialet var ett bra komplement till kursen.

Samtliga testsvar som kommenterade texten angav att studenterna tilltalades av textens nivå och formuleringar. Här följer några citat från studenter som testat materialet:

> “Det lättsamma sättet som texten var skriven på gjorde att man inte tappade intresset.”
>
> “Texten var i allmänhet bra och kul att läsa, gillade den lättsamma tonen.”
>
> “Nivån är väldigt bra och att ni ibland använder monolog, där ni ställer en fråga och sen svarar som ‘en vän’, gör att det känns mer begripligt och man får en känsla av att man pratar och diskuterar med en kompis om diverse ämnen. Detta ökar då förståelse enormt då det är annorlunda skrivet än hur till exempel föreläsare skriver sitt material.”

Testgruppen ansåg även att programmeringsövningarna var begripliga och låg på en bra nivå. Några citat från testgruppen angående övningarna följer här:

> “Övningarna känns begripliga och att ni visar vilket ‘out-come’ som efterfrågas hjälper en att förstå vad exakt som behöver göras.”
>
> “De övningarna jag gjorde kändes lagom svåra.”

Responsen från testgruppen innefattade även konkret kritik, exempelvis önskade flera studenter ett annat typsnitt för övningarna än för den övriga texten. Materialet har uppdaterats utifrån dessa önskemål och kommentarer.

8 Diskussion

Som nämns i inledningen har datateknologer svårt för kursen TSS och enligt examinatorerna beror detta på att de har svårt för matematiken, se *bilaga B*. Frågor man då kan ställa sig är vad som är orsaken till detta problem, huruvida problemet går att lösa helt, och om det verkligen är en bra metod att använda programmering för att lösa det?

Något som möjligen är en del av orsaken till problemet är när kursen TSS ges. Det nuvarande upplägget på datateknologernas studier är sådant att de läser ett antal kurser i matematik under sitt första år, knappt någon matematik under sitt andra år och sedan läser de TSS i årskurs tre. Det är alltså nästan ett år mellan matematikkurserna och TSS. Denna del av problemet kan i så fall lösas genom att till exempel ge TSS i årskurs två. Det är dock osäkert om detta är en tillräckligt stor del av problemet för att en förändring i kursupplägget skulle ge synbar effekt.

Under intervjun med examinatorerna framgick det att de tror att en stor del av problemet ligger i svårigheten att koppla den formella matematiken till dess praktiska tolkning.

Detta skulle i så fall kunna orsaka problem när datateknologer kommer i kontakt med matematiska läromaterial. Många matematiska texter är uppbyggda så att de har ett inledande fokus på formell matematik, med tyngd på matematisk korrekthet, och praktiska exempel och förklaringar följer först senare, om alls. Ett exempel på hur en matematisk lärotext kan vara upplagd är att man inleder med en formell definition, följd av ett matematiskt bevis, och till sist exempel. Denna typ av upplägg kan mycket väl fungera bra för studenter som är vana vid formell matematik men risken är att studenter som inte är det, till exempel datateknologer, tappar bort sig redan vid definitionen.

I de följande delavsnitt kommer projektets metod, resultat och möjlig framtida utveckling att diskuteras.

8.1 Metoddiskussion

För att utveckla ett väl fungerande läromaterial krävs goda kunskaper inom såväl det aktuella ämnet som didaktik. Projektgruppen består dock av fem studenter, vars erfarenhet inom området signal och systemlära var att fyra av dem läst kursen TSS. Studenternas tidigare kunskap i didaktik var också ytterst begränsad. På grund av detta lade projektgruppen en stor del av sin tid på nödvändig förberedande efterforskning. Detta lämnade mindre tid till utvecklingen av läromaterialet, vilket förorsakde att projektgruppen inte hade tillräckligt med tid att utarbeta ett lika utförligt material som den från början avsett. På grund av detta inkluderades inte Z-transform, Laplacetransform eller kursen Reglerteknik i materialet, vilket var den ursprungliga avsikten.

Man kan ställa sig frågan om det överhuvudtaget är en bra idé att låta studenter ta fram den här typen av läromaterial. En nackdel med detta är att studenterna

som skriver materialet behöver lägga en stor del av tiden på att lära sig ämnet och nödvändig didaktik tillräckligt bra, precis som vår projektgrupp har behövt göra. En följd av detta blir dock att studenterna får en djupare förståelse för ämnet samt får nya eller förbättrade kunskaper inom didaktik. Man kan rentav argumentera för att det är ett bra läromedel just att utveckla denna typ av läromedel.

En annan fördel med ett läromaterial som utvecklas av studenter är att dessa känner till målgruppen väl, eftersom de ingår i den. Studenterna har därför rimligtvis lättare att sätta sig in i vad som är svårt att förstå för en student än vad en expert inom ämnet har. Det är också möjligt att studenter har lättare för uttrycka sig på ett sätt som andra studenter kan förstå och ta till sig. Däremot kommer man inte ifrån att experterna har betydligt djupare kunskaper inom sitt ämne än studenterna och det är ytterst tveksamt att en grupp studenter skulle kunna ta fram ett läromaterial som är på samma nivå med lika goda insikter i ämnet som en expert. Det kan dock vara så att den här typen av studentutvecklade läromaterial är ett bra komplement till ämnesexperterna. En av kommentarerna från testgruppen, se avsnitt *7.2 - Testresultat* för hela sammanhanget, påpekade just att:

> "Att ni ibland använder monolog, där ni ställer en fråga och sen svarar som 'en vän', gör att det känns mer begripligt och man får en känsla av att man pratar och diskuterar med en kompis om diverse ämnen. Detta ökar då förståelse enormt då det är annorlunda skrivet än hur t.ex. föreläsare skriver sitt material."

Denna och andra kommentarer från testgruppen är starka argument för att denna typ av läromaterial kan fungera bra som komplement till annan undervisning.

En stor del av utmaningen med detta projekt har varit att det inte fanns någon uppenbar lösning på vad som skulle implementeras i programmeringskoden. För varje område som läromaterialet tar upp har projektgruppen behövt utvärdera vad som skulle ingå och vad som var möjligt att implementera på en nivå som studenterna i målgruppen kan förstå. Det har varit en ständig avvägning av att få med de mest vitala delarna i TSS och att implementera kod som är begriplig. *The Fastest Fourier Transform in the West* [16] är ett exempel på hur Fouriertransform kan implementeras relativt effektivt i Haskell, men problemet är att koden då hamnar på en nivå som överstiger förmågan hos många studenter i målgruppen. Kod av detta slag är alltså, oavsett hur effektiv den är, inte önskvärd i vårt läromaterial eftersom vi vill använda koden för att förklara ett annat ämne. Om koden är för svårläst riskerar vi att hamna i en situation där studenten varken förstår programmeringen eller ämnet den beskriver.

När man beslutat **vad** som ska implementeras uppstår frågan **hur** det ska implementeras. Som nämndes i avsnitt *6.4 - Implementationen av vårt DSL* finns det flera sätt att hantera detta. När man arbetar med signaler finns det två grundtyper; diskreta signaler och kontinuerliga signaler. Diskreta signaler kan implementeras relativt enkelt, medan kontinuerliga signaler blir betydligt mer komplicerat. Projektgruppen har under projektets gång granskat två olika metoder för att hanterar kontinuerlig tid i programmeringskoden. En variant är att approximera en kontinuerlig signal

med en diskret. Använder man väldigt små steg kan man få en relativt bra approximation på detta sätt, men det är fortfarande endast en approximation och därför blir resultatet inte fullständigt korrekt. En annan variant är att använda mönstermatchning, där funktioner definieras exakt för ett antal givnatypfall av insignaler som definieras var för sig. Detta ger mer korrekta svar, men inte samma möjlighet att sammanfatta området som en diskret approximation eftersom man definierar olika fall var för sig. Detta gör det å andra sidan möjligt att definiera typfall som studenten kan känna igen i kursen TSS.

Vad är då bäst i vårt fall, att använda en approximation som ger större möjlighet att sammanfatta kärnan i ämnet eller en mer matematiskt korrekt variant som använder formler som studenten kan känna igen? Detta är en fråga som projektgruppen inte har kunnat ta fram ett generellt svar på. Projektgruppen har, efter mycket diskussion, valt att använda båda varianter i läromaterialet. Vi ansåg att det fanns stora för- och nackdelar med båda lösningarna och att de argument som vägde tyngst berodde på vilket ämne som togs upp i det aktuella avsnittet. Inom vissa områden avgjorde gruppen att det gav större förståelse att sammanfatta och abstrahera, medan matematisk exakthet och typfall bedömdes vara viktigare inom andra områden.

Som beskrivs i avsnitt *3.5 - Didaktik* är en av utmaningarna när man utvecklar läromaterial att göra materialet tilltalande utan att det blir ren underhållning. Detta innebär en ständig balansgång när man utvecklar materialet. Å ena sidan vill man att materialet ska vara så informativt och effektivt som möjligt. Å andra sidan vill man fånga och behålla studentens intresse och uppmuntra till vidare inlärning. Vi har i vårt material strävat efter att beakta båda sidorna och försökt hålla en lättsam och lätt humoristisk ton utan att överskugga det faktiska innehållet. Mer om hur detta gått diskuteras i nästa avsnitt.

8.2 Resultatdiskussion

Den slutgiltiga produkten uppfyller alla delar av produktspecifikationen med ett undantag. Ambitionen var att läromaterialet skulle täcka in hela kursen TSS. I nuläget saknas dock avsnitt som beskriver Laplacetransform och Z-transform. Som nämns i avsnitt *6.2 - Läromaterialets struktur* prioriterades detta bort på grund av tidsbrist i slutet av projektet. Projektgruppen hamnade i läget att de kunde fokusera på antingen Fouriertransform eller Laplacetransform och Z-transform. Anledningen till att gruppen valde att fokusera på Fouriertransform var att den är nära besläktad med Laplacetransform och Z-transform. Ur programmeringssynpunkt var det även svårt att implementera Laplacetransform och Z-transform utan att först implementera Fouriertransform. Det finns emellertid utkast till avsnitt om Laplacetransform och Z-transform som kan utvecklas i framtiden.

Som beskrivs i avsnitt *6.6 - Test av läromaterialet* ställde sig testgrupppen väldigt positiv till läromaterialet. Det är dock värt att påpeka att testgruppen bestod av en relativt liten grupp, totalt 13 personer. Även om testgruppens svar är en positiv

indikation är det därför svårt att dra några definitiva slutsatser om huruvida vårt läromaterial uppnår sitt mål i nuläget.

Vidare är det så att flera personer i testgruppen går i samma årskurs och på samma program som flera av medlemmarna i projektgruppen. Återkopplingen kunde av praktiska skäl heller inte vara helt anonym. Det är därför inte orimligt att anta att vissa medlemmar i testgruppen kan ha varit partiska i sin bedömning och att detta kan ha påverkat resultatet.

För en bättre utvärdering av läromaterialet krävs att fler studenter inom målgruppen använder sig av det i samband med kursen TSS. Det kommer därför dröja innan man kan se det verkliga resultatet. Detta kan dock vara svårt att utvärdera även då eftersom det finns många andra faktorer som kan påverka. Om en ökning i antalet godkända studenter i kursen observeras i framtiden är det svårt att säga med säkerhet i vilken utsträckning denna förändring beror på vårt läromaterial.

8.3 Den sociala dimensionen

Ett läromaterial som lyckas få datastudenter att *förstå* innehållet i en kurs, som de annars uppfatta som svår, kommer att leda till positiva påverkan i den sociala dimensionen.

Det direkta resultatet från ett väl fungerande läromaterial är att andelen godkända studenter i kursen höjs. Detta skulle i sin tur medföra att välbefinnandet höjs för studenterna som annars inte skulle klara kursen. Det skulle också möjligen medföra en minskad oro hos de studenter som ännu inte läst kursen i fråga, då den skulle förefalla mindre svår om det är fler studenter som får godkänt i kursen.

Det långsiktiga målet för projektet DSLs of Math, som vårt projekt baseras på, är att minska gapet mellan matematik och programmering. Detta skulle kunna öppna upp för fler möjligheter att tackla framtida utmaningar, eftersom det skulle utbildas fler ingenjörer som inte är begränsade till enbart ett tankesätt.

Läromaterial som vårt är då i bästa fall ett *socialt stöd* (social support) som hjälper studenterna att få en bättre förståelse inom båda domänerna. Detta skulle i så fall kunna både förbättra studenternas välbefinnande och även vara till nytta för samhället där de framtida ingenjörerna kommer att arbeta.

Det förefaller dock ytterst osannolikt att endast ett läromaterial kan åstadkomma en tillräckligt stor effekt för att orsaka alla dessa positiva konsekvenser. Dock förefaller det utifrån våra testresultat och andra liknande studier som om ett projekt som vårt kan vara ett bidrag till att minska gapet mellan matematik och programmering och bidra till dessa konsekvenser.

8.4 Framtida utveckling

Vad återstår då att utveckla i framtiden? En möjlighet är givetvis att vidareutveckla det befintliga läromaterialet. Till att börja med skulle man kunna komplettera materialet med avsnitt om Laplacetransform och Z-transform. Det finns, som tidigare

nämnts, påbörjade utkast för dessa och om de färdigställs täcker läromaterialet in hela kursen TSS i stora drag.

En annan vidareutveckling av materialet skulle kunna vara utökad implementation av funktioner i kontinuerlig tid. Som vi tidigare nämnt i *8.1 - Metoddiskussion* är detta ett relativt komplext problem som i vårt fall löstes med en kompromiss. Detta är dock ett område där djupare studier säkerligen skulle kunna ge bättre implementationer.

Om fler tester genomförs skulle materialet eventuellt kunna förbättras ytterligare med dessa tester som underlag. En större testgrupp och anonyma svar skulle kanske kunna ge fler och mer tillförlitliga resultat. Det skulle också vara värdefullt att testa produkten tillsamans med kursen TSS.

Det skulle även kunna vara intressant att gå igenom materialet tillsammans med experter inom TSS samt Didaktik. Det är mycket möjligt att experterna har en annan syn på materialet än studenterna har och kan ge andra förslag på hur materialet kan utvecklas.

I vårt projekt valde vi att fokusera på kursen TSS men som beskrivs i bakgrunden är den bara en av problemkurserna på Datatekniska programmet. En möjlighet skulle vara att i framtiden ta fram ett liknande läromaterial även till kursen Reglerteknik, vilket var vår ursprungliga avsikt. Ett sådant läromaterial skulle då kunna använda sig av delar av den programmeringskod som tagits fram i vårt material, eftersom Reglerteknik till viss del bygger på innehållet i TSS.

Även andra kurser än de som tas upp i denna rapport skulle kanske kunna gagnas av liknande läromaterial. Som vi nämner i inledningen och *3.6 - Relaterad forskning* finns liknande problem även inom andra områden.

Hela läromaterialet finns på `https://github.com/DSLsofMath/BScProj` under licensen *Creative Commons erkännande, dela lika*. Det är med andra ord fritt fram för vem som helst att arbeta vidare med materialet förutstatt att man anger upphovsman och använder samma licens för vidareutvecklingar som bygger på vårt material.

9 Slutsatser

Projektets mål var att ta fram ett kompletterande läromaterial till kursen TSS inspirerat av DSLs of Math som uppfyller produktspecifikationen som beskrivs i avsnitt *5.1 - Produktens struktur*.

Projektgruppen har utifrån detta tagit fram ett material som finns samlat på en hemsida. Materialet innehåller förklarande text och exempel, bilder samt programmeringsövningar. Det består av 4 delavsnitt som i stora drag täcker in innehållet i kursen TSS, med undantag för Laplacetransform och Z-transform.

Enligt testgruppen är texten underhållande och informativ och övningarna ligger på en lagom svår nivå. På grund av testgruppens ringa storlek är det svårt att dra några definitiva slutsatser, men testresultaten indikerar att materialet uppfyller de krav som ställs i produktpecifikationen. Något annat som gör den faktiska kvalitén på läromaterialet svår att bedöma är att det ännu inte har använts i samband med kursen.

Sammanfattningsvis har projektgruppen alltså tagit fram ett läromaterial som uppfyller kraven i produktspecifikationen, med undantag av ett saknat delavsnitt. Testresultaten har varit positiva men är inte tillräckliga för att dra definitiva slutsatser. Hur väl detta material fyller sin funktion kommer förhoppningsvis att visa sig i framtiden när det används av studenter i målgruppen i samband med kursen TSS.

Referenser

[1] G. J. Sussman och J. Wisdom, "The role of programming in the formulation of ideas", 2002.

[2] C. Wells, "Communicating mathematics: Useful ideas from computer science", *The American mathematical monthly*, vol. 102, nr 5, s. 397–408, 1995.

[3] C. Ionescu och P. Jansson, "Domain-specific languages of mathematics: Presenting mathematical analysis using functional programming", i *Proc. 4th Int. Workshop on Trends in Functional Programming in Education*, ser. EPTCS, Open Publishing Association, 2015.

[4] A. Bränberg, H. Gulliksson och U. Holmgren, *Didaktik för ingenjörslärare : Konsten och glädjen med att utbilda ingenjörer*, 1. utg. Lund : Studentlitteratur, 2013, ISBN: 9789144089270.

[5] J. Hughes, "Why functional programming matters", *The computer journal*, vol. 32, nr 2, s. 98–107, 1989.

[6] J. Svenningsson och E. Axelsson, "Combining deep and shallow embedding for EDSL", i *Trends in Functional Programming*, Springer, 2012, s. 21–36.

[7] J. M. Keller, *Motivational design for learning and performance : The ARCS model approach*, 1. utg. New York : Springer, 2009, ISBN: 9781441965790.

[8] Chalmers tekniska högskola. (2016). DAT325 - matematikens domänspecifika språk, URL: `https://www.student.chalmers.se/sp/course?course_id=24179`.

[9] ——, (2016). Statistik över kursresultat, URL: `http://document.chalmers.se/doc/00000000-0000-0000-0000-00001C968DC6`.

[10] ——, (2014). Ere102 - reglerteknik, URL: `http://document.chalmers.se/doc/b1ebfabc-b2c1-41a3-bf22-35711526aae8`.

[11] K. Azad. (2016). Betterexplained, URL: `http://betterexplained.com/` (hämtad 2016-02-15).

[12] M. Lipovača, *Learn You a Haskell for Great Good!* No Starch Press, 2011, ISBN: 9781593272838.

[13] G. Ejlertsson, *Enkäten i praktiken : En handbok i enkätmetodik*, 2. utg. Lund : Studentlitteratur, 2005, ISBN: 9144031645.

[14] J. Trost, *Enkätboken*, 2. utg. Lund : Studentlitteratur, 2001, ISBN: 9144018169.

[15] J. Conway, *Functions of one complex variable*. New York: Springer-Verlag, 1978, ISBN: 0387903283.

[16] J. Brown. (2016). Fastest fourier transform in the west, URL: `https://hackage.haskell.org/package/fft-0.1.8.3/docs/Math-FFT.html`.

A Bilaga 1

Denna bilaga är ett utdrag ur den enkätundersökning som genomfördes av kandidatgruppen under våren 2016 via *webbenkater.com*. Bilagan innehåller endast de delar av undersökningen som bedömts som relevanta för rapporten och är även tömd på all information som kan länkas till enskilda individer.

Observera att antalet deltagare varierar markant. Detta beror på att fråga 1 och 9 riktade sig även till studenter som ska läsa kursen nästa år, medan övriga frågor endast riktar sig till studenter som redan läst kursen.

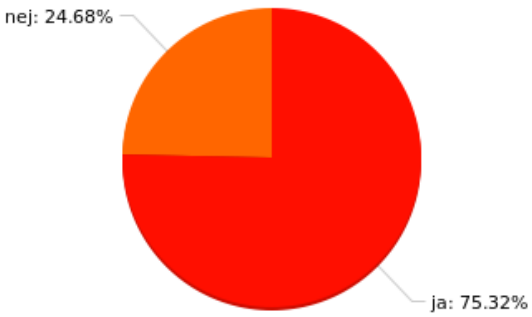
Samtliga bilder i bilagan kommer från *webbenkater.com*.

Fråga 1 - Har du läst kursen “Transformer, signaler och system”?

Antal deltagare: 77

58 (75.3%): ja

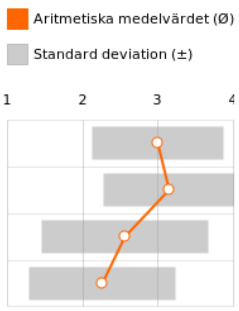
19 (24.7%): nej



Fråga 9 - Hur väl upplever du att du behärskar följande områden?

Antal deltagare: 73

	Behärskar inte alls (1)		Behärskar till viss del (2)		Behärskar någorlunda (3)		Behärskar väl (4)			
	Σ	%	Σ	%	Σ	%	Σ	%	Ø	±
Matematisk analys i en...	4x	5,48	16x	21,92	30x	41,10	23x	31,51	2,99	0,87
Komplexa tal	3x	4,11	14x	19,18	26x	35,62	30x	41,10	3,14	0,87
Komplexa exponentialf...	16x	21,92	19x	26,03	19x	26,03	19x	26,03	2,56	1,11
Elektriska nät	18x	24,66	27x	36,99	19x	26,03	9x	12,33	2,26	0,97

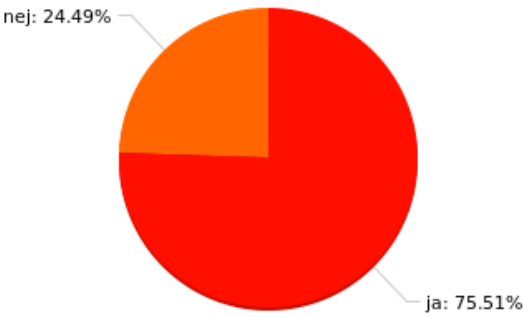


Fråga 11 - Upplevde du att du hade tillräckliga förkunskaper för kursen "Transformer, signaler och system"?

Antal deltagare: 49

37 (75.5%): ja

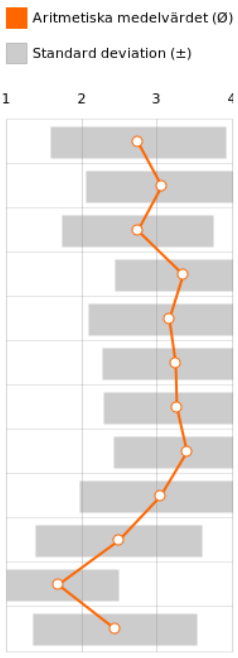
12 (24.5%): nej



Fråga 12 - Hur väl upplever du att du behärskar följande områden inom "Transformer, signaler och system"?

Antal deltagare: 48

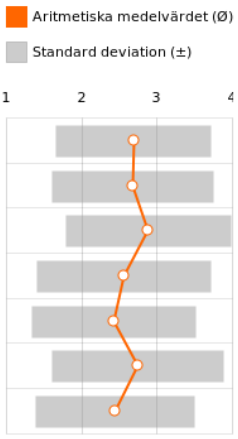
	Behärskar inte alls (1)		Behärskar till viss del (2)		Behärskar någorlunda (3)		Behärskar väl (4)			
	Σ	%	Σ	%	Σ	%	Σ	%	Ø	±
Fourierserier	9x	18,75	12x	25,00	9x	18,75	18x	37,50	2,75	1,16
LTI-system och deras ...	5x	10,42	7x	14,58	16x	33,33	20x	41,67	3,06	1,00
Faltning	7x	14,89	10x	21,28	18x	38,30	12x	25,53	2,74	1,01
Laplacetransform	4x	8,33	2x	4,17	15x	31,25	27x	56,25	3,35	0,91
Z-transform	7x	14,58	3x	6,25	13x	27,08	25x	52,08	3,17	1,08
Impulssvar	5x	10,42	3x	6,25	15x	31,25	25x	52,08	3,25	0,98
Stegsvar	5x	10,42	3x	6,25	14x	29,17	26x	54,17	3,27	0,98
Överföringsfunktion	5x	10,42	2x	4,17	10x	20,83	31x	64,58	3,40	0,98
Frekvenssvar	7x	14,58	5x	10,42	15x	31,25	21x	43,75	3,04	1,07
Fourierrepresentation...	11x	23,40	13x	27,66	12x	25,53	11x	23,40	2,49	1,10
Parsevals formel	24x	50,00	16x	33,33	7x	14,58	1x	2,08	1,69	0,80
Diskret Fouriertransfor...	11x	22,92	16x	33,33	10x	20,83	11x	22,92	2,44	1,09



Fråga 13 - Hur väl upplever du att du kan följande delmoment inom "Transformer, signaler och system"?

Antal deltagare: 48

	Inte alls (1)		Till viss del (2)		Till stor del (3)		Bra (4)			
	Σ	%	Σ	%	Σ	%	Σ	%	Ø	±
Identifiera och ge exemp...	6x	12,50	17x	35,42	11x	22,92	14x	29,17	2,69	1,03
Ge en sammanfattning a...	8x	17,02	12x	25,53	14x	29,79	13x	27,66	2,68	1,07
Beräkna utsignalen från li...	7x	14,58	11x	22,92	11x	22,92	19x	39,58	2,88	1,10
Beskriva signaler i frekve...	10x	20,83	16x	33,33	7x	14,58	15x	31,25	2,56	1,15
Beskriva samplingsproce...	12x	25,00	14x	29,17	12x	25,00	10x	20,83	2,42	1,09
Utifrån ett linjärt systems...	7x	14,58	17x	35,42	5x	10,42	19x	39,58	2,75	1,14
Göra en grundläggande t...	10x	20,83	17x	35,42	11x	22,92	10x	20,83	2,44	1,05



Fråga 14 - Upplevde du att du hade tillräckliga förkunskaper för kursen "Reglerteknik"?

Antal deltagare: 46

20 (43.5%): ja

26 (56.5%): nej

Fråga 19 - Har du några övriga kommentarer om innehållet i kursen "Transformer, signaler och system"?

Antal deltagare: 10

- Innehållet som täcks är relevant.
- Sjukt bra.
- Lägg nivån lägre så studenterna förstår bättre. Som det var när jag klarade kursen satt jag bara och nötte tentor. Kommer ihåg ytterst lite av innehållet nu. Känns inte som det ska vara så.
- Lägg mer tid på att _pedagogiskt_ förklara motivationen bakom de olika transformerna. Själva tillämpningen av dem kom först långt efter att man gått igenom teorin (alltså se till att folk kommer igång tidigt med att räkna).
- Innehållet är bra, och det känns väl kopplat till verkligheten.
- Det var ofta som man fick verktyget innan man förstod vad det gjorde. Jag vill veta målet innan jag får reda på metoden.
- Den var bra.

Fråga 20 - Har du några övriga kommentarer om kursen "Transformer, signaler och system" i allmänhet?

Antal deltagare: 15

- Rolig kurs!
- Mycket givande, rolig och stabil kurs
- Kursen ger en väldigt bra förberedelse inför reglertekniken. För där är det hjälpsamt att ha förståelse för var transformerna kommer ifrån.
- Lära ut koncepten mer generellt och minska focus på svårare matte omvandlingar.
- Sjukt bra kurs.
- Hade inga egentliga problem med kursen. Anser inte att något extra material behövs för att lära sig innehållet.
- Bra kurs som inte var svårare än något annat vi läst.
- Bra o lätt kurs. Ser inte varför alla tycker den e så svår...

Fråga 21 - Har du några övriga kommentarer om innehållet i kursen "Reglerteknik"?

Antal deltagare: 13

- Cool kurs, men lite väl spretig
- Mer datarelaterat kursmaterial hade underlättat enormt.
- För oteoretiskt
- Fysik-uppgifterna var alldeles för svåra. Otillräckliga förkunskaper?
- Samma som TTS, lägg nivån lägre.
- Innehållet är det inget större fel på.
- Borde anpassas mer till data. Mer programmeringsaktiga problem, mindre fysik/matte
- Fysiken var inte svår, den var ovan. Jag har koll på fysik när jag går en fysikkurs, men för min del är fysik en färskvara. Jag försökte läsa boken, men den var svår att förstå.
- Inte fel på materialet, fel på studenterna

Fråga 22 - Har du några övriga kommentarer om kursen "Reglerteknik" i allmänhet?

Antal deltagare: 22

- Kursen är inte svår, teknologerna är dåliga.
- Också en rolig kurs!
- Icke-teoretisk, förklaras dock delvis av reglerteknikens mer experimentella karaktär
- Kursen är för tung och går iväg på sidospår.
- Mer omfattande laborationer som täcker större delar av kursinnehållet. De nuvarande är för få/små/simpla.
- Kombinera nyckelkoncepten med TSS och gör en kurs av det för att lämna utrymme till annat som är användbart. Man behöver inte 15hp för att lära sig det som är användbart i båda kurserna (signal tolkning och transformation samt reglerade system)..
- Hade inga egentliga problem med kursen. Anser inte att något extra material behövs för att lära sig innehållet.
- Man undrar hur kurser med så pass hög procent underkända varje år får vara kvar utan att det händer något. TIF085 är en annan sådan kurs.
- Inte så svår som folk skrämt upp en till att tro. Men det krävs en stor personlig insats då kursmaterialet inte alltid är så lättillgängligt (onödigt krångligt, ger ingen intuitiv känsla).
- Problemet med den här kursen är helt enkelt att tycker den är tråkig. Innehållet är svårare än tidigare kurser, samt datastudenter är vana vid att lämna mycket arbete till sista minut. I den här kursen gör det att man hamnar efter direkt och inte hänger med på föreläsningarna. Då tappar elever intresset, lägger ännu mindre tid på kursen, och missar tentan. Jag tror att något sånt här sätt att anpassa materialet för dataintresserade elever skulle hjälpa enormt med att hålla intresset uppe.
- Sjukt intressant, inte alls svår som man hade hört
- Den ligger för långt i tiden ifrån mattekurserna.

B Bilaga 2

Denna bilaga är en sammanfattning av den intervju projektgruppen höll med Bo Egardt och Ants Silberberg. Bo och Ants har under flera år varit både lärare och examinatorer i Reglerteknik respektive TSS på Chalmers. Sammanfattningen är granskad och godkänd av både Bo och Ants.

Möte med Bo Egardt och Ants Silberberg 10:e februari

En sammanfattning

Vad upplevde du var datastudenternas huvudsakliga problem med ämnet och varför tror du att just det här ämnet är så svårt för dem?

- Datastudenter är, generellt sett, inte tillräckligt tränade på att räkna. De saknar vana vid det matematiska hantverket.

- Datastudenter har ofta svårt att se teorin bakom matematiken, som till exempel signaler i kursen TSS. De har svårt att förstå vad matematiken står för.

- Datastudenter har ofta allmänt svårt att se bortom och tolka det formella i matematiken istället för att bara se formler.

- Datastudenter har ofta svårt att röra sig mellan formell matematisk abstraktion och tolkningen av denna. Det saknas några steg i abstraktionstrappan.

- Generellt sett lägger datastudenter 95% av sin kraft på matematiken och endast 5% på tolkningen, istället för tvärtom vilket skulle vara önskvärt. Studenterna behöver lägga sin energi på innebörden för att förstå ämnet.

Allmänt

- Det är relativt lätt att behandla diskret tid inom programmering. Dock missar man en stor del av ämnet genom att inte också betrakta kontinuerlig tid eftersom den fysikaliska modelleringen då bortfaller.

- Man tjänar mycket på att identifiera temat i kurserna. Abstraherar man detta ser man att det finns mycket gemensamt för delområdena och de kan många gånger närmast betraktas som olika varianter av samma sak.

C Bilaga 3

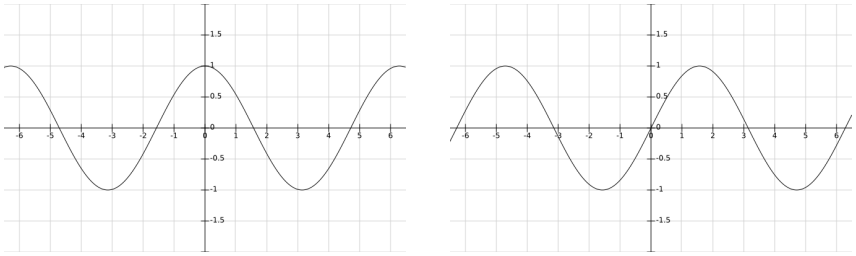
Denna bilaga är ett utdrag ur det läromaterial projektgruppen utvecklat. Utdraget beskriver udda och jämna signaler och är hämtat från avsnitt 2 i läromaterialet.

Udda och Jämna Signaler

Eftersom signaler kan betraktas som matematiska funktioner kan vi säga att vi har udda och jämna signaler, precis som vi har jämna och udda funktioner. En funktion kan vara jämn, udda eller inget utav dem.

En funktion f är jämn om $f(x)$ är samma som $f(-x)$. I den vänstra bilden här nedan ser du ett exempel på en jämn funktion, $f(x) = \cos(x)$. Notera att vilken punkt du än tittar på i figur 6a här nedan så gäller alltid $f(x) = f(-x)$.

En funktion f är udda om $f(-x)$ är samma som $-f(x)$. I den högra bilden nedan ser du ett exempel på en udda funktion , $f(x) = sin(x)$. Notera att vilken punkt du än tittar på i figur 6b här nedan så gäller alltid $f(-x) = -f(x)$.



(a) Jämn funktion

(b) Udda funktion

Figure 1

Alltså:

Jämn funktion: $f(x) = f(-x)$

Udda funktion: $f(-x) = -f(x)$

Varför är det då bra att veta om en signal är jämn eller udda? Mer om det kommer i avsnittet om Fourierserier!

Kryssfråga 8: Är funktionen $f(t) = 0.5\sin(2t)$ här nedan jämn, udda eller ingetdera?

a) Jämn

b) Udda

c) Ingetdera

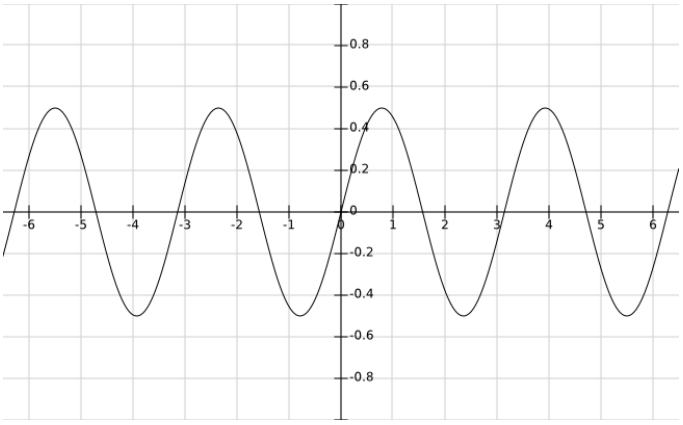


Figure 2: funktionen $f(t) = 0.5\sin(2t)$

Kryssfråga 9: Är funktionen $f(t) = 3\cos(0.25t)$ här nedan jämn, udda eller ingetdera?

a) Jämn

b) Udda

c) Ingetdera

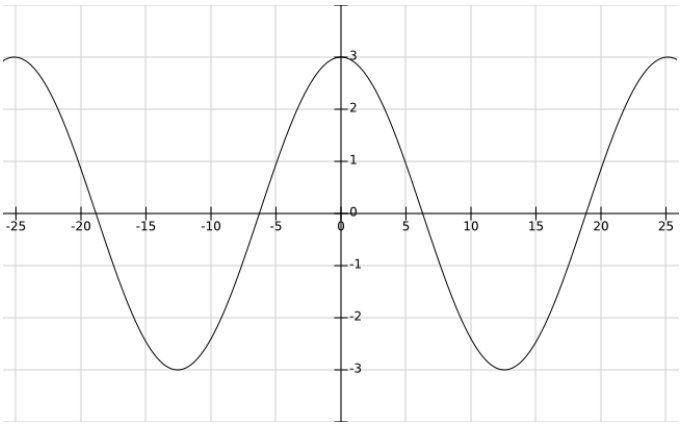


Figure 3

Då ska vi se hur man kan implementera det här! En funktion som avgör om en signal är udda kan se ut så här:

```
contIsOdd :: ContTimeFun -> ContTime -> Double -> Bool
contIsOdd signal step limit = and $ map testPair $ zip xs ys
    where testPair (a,b) = a ~= (-b)
          xs = map signal [0, step .. limit]
          ys = map signal [0,-step .. (-limit)]
```

Övning 7: Implementera motsvarande funktion som undersöker om en signal är jämn.

Test: Lös de två nedanstående uppgifter och kontrollera sedan svaret med hjälp av programmet.

Test 1: En signal har funktionen: $3 * \sin(6t)$

Vad är signalens period?

Är signalen udda eller jämn?

Test 2: En signal har funktionen: $\frac{1}{2}\cos(t)$

Vad är signalens period?

Är signalen udda eller jämn?

Testa gärna fler signaler!