Populationsdynamik

Slutprojekt inom Numeriska Metoder SF1546

Grupp: Populationsdynamik 9

Noel Karlsson Johansson

20000519-0517

noelkj@kth.se

Tawsiful Islam

20001110-2035

tawsiful@kth.se

**Innehållsförteckning**

[**1. Inledning** 2](#_Toc38725274)

[**2. V(t) konstant slutvärde** 4](#_Toc38725275)

[Frågeställning 4](#_Toc38725276)

[Metod 4](#_Toc38725277)

[Resultat 4](#_Toc38725278)

[**3. T1 när växter uppnår 95 % av slutvärde** 4](#_Toc38725279)

[Frågeställning 4](#_Toc38725280)

[Metod 4](#_Toc38725281)

[Resultat 5](#_Toc38725282)

[**4. Växters och skadedjurs konstant slutvärde** 6](#_Toc38725283)

[Frågeställning 6](#_Toc38725284)

[Metod 6](#_Toc38725285)

[Resultat 6](#_Toc38725286)

[**5. Växters och skadedjurs population mellan T1 och T2 =1.5** 7](#_Toc38725287)

[Frågeställning 7](#_Toc38725288)

[Metod 7](#_Toc38725289)

[Resultat 8](#_Toc38725290)

[**6. Växters, skadedjurs & rovdjurs konstant slutvärde** 9](#_Toc38725291)

[Frågeställning 9](#_Toc38725292)

[Metod 9](#_Toc38725293)

[Resultat 10](#_Toc38725294)

[**7. Växters, skadedjurs & rovdjurs population tills T3 =3** 11](#_Toc38725295)

[Frågeställning 11](#_Toc38725296)

[Metod 11](#_Toc38725297)

[Resultat 11](#_Toc38725298)

[**8. Årlig besprutnings påverkan på populationerna** 12](#_Toc38725299)

[Frågeställning 12](#_Toc38725300)

[Metod 12](#_Toc38725301)

[Resultat 13](#_Toc38725302)

[**9. Modellens känslighet** 13](#_Toc38725303)

[Frågeställning 13](#_Toc38725304)

[Metod 14](#_Toc38725305)

[Resultat 14](#_Toc38725306)

[**10. Tid för att skörda utan besprutning** 14](#_Toc38725307)

[Frågeställning 14](#_Toc38725308)

[Metod 14](#_Toc38725309)

[Resultat 15](#_Toc38725310)

[**11. Tid för att skörda med besprutning** 16](#_Toc38725311)

[Frågeställning 16](#_Toc38725312)

[Metod 16](#_Toc38725313)

[Resultat 17](#_Toc38725314)

[**12. Optimala tid för att skörda med besprutning** 18](#_Toc38725315)

[Frågeställning 18](#_Toc38725316)

[Metod 18](#_Toc38725317)

[Resultat 18](#_Toc38725318)

[**13. Eget arbete** 18](#_Toc38725319)

[**14. Bilagor** 19](#_Toc38725320)

# Inledning

Vi fick givet att 100 exemplar av en växtart *V* planteras på en ö. Därefter över tid kommer det skadedjur *S* samt rovdjur *R*. Det vi skulle undersöka var arternas population i ekosystemet och hur de påverkas av olika faktorer såsom andra djur, besprutning och skörd.

I början av denna uppgift skulle vi beräkna vilket konstant slutvärde modellen för växtens population uppnår när inga djur finns på ön. Dessutom efterfrågades det vilken tid T1 populationen uppnår 95 % av konstant slutvärdet. Modellen kunde lösas analytiskt för att hitta dess konstant slutvärde. En ungefärlig tid hittades med Runge-Kutta metoden, med noggrannhetsordning 4, kring 95 % av värdet och därefter med en andragradsinterpolation kring den tiden. Den andragradsekvationen som fick löstes med PQ-formeln för att få en exakt lösning. Konstant värdet blev 882352 växter och tiden T1 blev 0.80196686 år.

Sedan fick vi givet en olinjär modell för växten och skadedjurets population som berodde av varandra från T1. Med denna modell skulle vi beräkna vilket konstant slutvärdet populationerna uppnår. Dessutom ska vi kolla ifall populationen stabiliserade sig nära värdet vid T2 = 1.5 och hur mycket det avvek. Med Newtons metod för flera variabler löste vi att *V* och *S* blev 101069 respektive 603. Vid T2 kom vi fram till att båda hann stabilisera sig och avvek med 4.05 respektive 4.83 promille från konstant slutvärdena funna med Newtons metod.

Efter T2 introduceras rovdjuren och differentialekvationen för skadedjuren förändrades lite då skadedjuren och rovdjurens population berodde av varandra medan modellen för växten var oförändrad. Återigen skulle konstant värdena räknas ut med Newtons metod som alla tre uppnår när tiden går mot oändligheten. Dessutom jämförde vi värdena vid T3 = 3 för alla tre arter. Värdena räknas ut med Runge-Kutta metoden. Vi fick fram att modellerna närmade sig konstant slutvärdena 416756, 359 samt 31 för V, S respektive R. Populationerna vid T3 blev 376380, 382 och 28 för V, S respektive R. Värdena skiljdes med 9.69 %, 6.11 % samt 9.68 % från konstant värdena för de tre olika arterna.

Vid T3 bestämmer sig öborna att ha en årlig besprutningskampanj som dödar 70 % av skadedjuren samt 20 % av rovdjuren vid besprutning. Vi skulle undersöka hur växternas population ser ut ett år före samt ett år efter besprutningskampanj och ifall det var gynnsamt. Vi genomförde Runge-Kutta metoden för att se hur populationen förändrades över tid samtidigt som vi ändrade djurens population vid årsskiftet T = 4, 5, 6, … när besprutningen sker. Genom att jämföra värdena som uppnås vid med och utan besprutning kom vi fram till att det är gynnsamt att skörda under en viss period på året då populationen överstiger vår referenslinje utan besprutning. Därefter understiger det referenslinjen.

Vi skulle dessutom undersöka vilken koefficient i alla differentialekvationer som påverkade V:s population som mest. Vi ändrade varje koefficient i taget till ett värde som var 120 % av det existerade koefficient, dvs. 20 % ökning eller minskning av koefficienten. När vi undersökte antalet växter mellan T = 1.5 och 3 utan besprutning med Runge-Kutta och begynnelsevärdena V och S hade vid T = 1.5 och R hade 2. Kom vi fram till att vid T = 3 var växternas population som störst om koefficienten i d*V*/d*t* byttes till ett värde som var 120 % av 15. Vi drog en generell slutsats att modellen i allmänhet inte var alltför känslig. Endast förändringar i två koefficienter gav stora avvikande resultat från värdet med alla koefficienter oförändrade.

Som utvidgning till uppgiften skulle vi undersöka hur populationerna påverkades beroende på när skörden sker. Skörden sker τ år in på året. Först skulle vi undersöka hur populationerna ser ut mellan T = 0 till 8 om man skördar vid T = 4 + τ, T = 5 + τ, T = 6 + τ osv. Genom att plotta hur stor skörden var beroende på τ år 7 kom vi fram till att skörden var lika stor oavsett τ. Detta skulle undersökas utan årlig besprutning.

Däremot undersöker man situationen med årlig besprutning spelade det roll, vilket var nästa deluppgift. Resultatet fick vi fram på liknande sätt. Skörden var gynnsamt mellan τ för 0.01 och 0.12 enligt grafen vi fick tillsammans med linjen för utan besprutning.

Den sista frågeställningen vi skulle besvara var att hitta den optimala τ när skörden är som störst år 7. Vi använde principen om gyllene snittet för optimering. För det tog vi hjälp av en beskrivning för algoritmen av Gerd Eriksson i *Numeriska algoritmer med Matlab*. Vi fick fram att skörden var som störst om man skördar vid τ = 0.0563.

# V(t) konstant slutvärde

## **Frågeställning**

100 exemplar av en växt planteras år 0 på ön. Vi fick givet att följande differentialekvation beskriver växtens fortplantning över tid, tidsenheten är år.

(Ekv. 1)

Växten växer till ett konstant slutvärde när t ökar som vi ska hitta.

## **Metod**

Vi kunde identifiera att detta är en separabel differentialekvation och kunde lösas analytiskt. När ett värde är konstant är derivatan . Därmed faktoriserade vi ut *V* för att lösa *V* i de båda faktorerna när . Därmed tog vi hjälp av nollfaktorlagen.

## **Resultat**

Vi fick fram att ekvationen var lika med 0 när och . Det är mer logiskt att när den uppnår sitt konstantvärde eftersom det ska finnas växter. Därmed uppnås slutvärdet . Däremot kan vi säga att maximalt 882352 växter kan växa för vi har hela plantor och det överstiger inte det värdet.

# T1 när växter uppnår 95 % av slutvärde

## **Frågeställning**

Nästa uppgift vi skulle lösa var att få fram vid vilken tid T1 uppnår populationen 95 % av sitt slutvärde.

## **Metod**

Vi räknade fram att slutvärdet var och 95 % av det blir 838235,29412. Vi började först med att använda av oss Runge-Kutta med noggrannhetsordning 4 som finns i filen *rk4.m*, vilket var en effektiv numerisk metod för differentialekvationer. Vår differentialfunktion bestod av Ekv. 1 och vi hade gett begynnelsepopulationen för växterna som var *v* = 100 och starttiden är *t0* = 0. Filen bestod av två funktioner *rk4()* och *calcRK().*

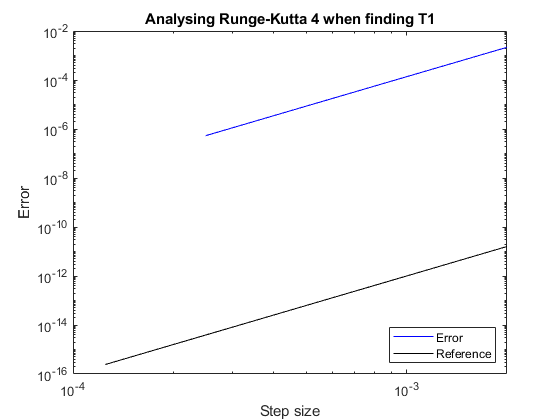
*calcRK()* genomförde Runge-Kutta med en inmatning för populationen *V*, funktionen *fun* som var differentialekvationen samt steglängden som bestämdes till *h =* 0.002. Funktionen *rk4()* fick inmatningarna starttid *t0*, startpopulationen *v*, funktion *fun*, tidsstopp eller maximal värde *limit* samt ifall tiden är känd med parametern *s* = 1 eller 2. Om tiden var okänd, som i detta fall, var parametern *s*=1 och körde den if-satsen som gällde för det. För denna if-sats i *rk4()* hade vi ett maximal värde *limit* vi räknade fram till (se bilaga 1). While-loopen vi använde slutade tills *v* överstiger 99 % av slutvärdet (vår *limit*), eftersom det inte går att överstiga 100 % av slutvärdet. Datasamlingen inkluderade antal växter vid 95 % av slutvärdet.

När vi hade alla datapunkter för växtpopulationen från t = 0 till t för 99 % av slutvärdet skickades denna matris *y* tillsammans med 95 % av slutvärdet *v95* till funktionen *interpolT1()* som sökte exakta T1 vid *v95* med hjälp av kvadratisk interpolation. För att interpolera gjorde vi en naiv ansats med tre ekvationer med tre punkter för att göra ett andragradspolynomanpassning (se bilaga 2). De tre punkter vi tog var den närmaste approximativa värdet för *v95* samt punkterna före och efter den. Ekvationen löstes med Matlabs egna anpassade algoritm för gausseliminering och gav oss koefficienterna som var med i andragradpolynomet.

Polynomet löstes sedan analytiskt med PQ-formeln på Matlab när polynomet var lika med *v95* och tog det T1 som var närmast de approximativa tiden för *v95*. En andragradspolynom användes för att den har en bättre noggrannhet än linjär och kan lösas enkelt analytiskt för T1.

## **Resultat**

Det T1 vi fick fram med hjälp av interpolation är 0.80196686 som är tiden när växtpopulationen uppnår 95 % av sitt slutvärde. Det var en kort tid för ett högt värde. Men vi undersökte noggrannhetsordningen för Runge-Kutta och ifall den var 4 enligt teorin. Med hjälp av koden i *reliabilityRK()* körde vi Runge-Kutta med samma kod från *rk4()* och halverade steglängden för varje gång för att kolla felet i svaret. Vi såg att den åttonde decimalen skiljde sig när steglängden halverades och osäkerheten ligger där. När vi jämförde kvoten mellan två fel där felet var mellan två approximativa värden för samma tid, men olika steglängd som skiljdes med faktor 2, närmade sig kvoten mot 16 vid mindre steglängd (se bilaga 3, rad 71). Dock vid mycket korta steglängder blev kvoten annorlunda som vi antog berodde på trunkeringsfel. Detta bekräftades med en loglog-graf med en linje som var felen mot steglängden och en referenslinje som var steglängden upphöjt till fyra mot steglängden (se bilaga 3, rad 80).



**Figur 1:** Figuren visar hur felen förhåller sig till steglängden och ifall den är parallell med referenslinjen som följer teorin för Runge-Kutta 4.

Båda linjerna var parallella mot varandra (se figur 1), vilket innebar att vår *rk4()* följde teorin och var noggrann. Jämför man med osäkerheten i svaret på tiden T1 som vi fick drog vi också slutsatsen att valet av steglängden inte påverkade svaret alltför mycket. Storleken av felet är dessutom mindre än en dag, vilket är också anledningen bakom vår slutsats.

# Växters och skadedjurs konstant slutvärde

## **Frågeställning**

Vid T1 kommer det två stycken skadedjur *S*, vi kan säga att det är möss, och befinner sig på ön. Växternas och mössens population beror av varandra och beskrivs med följande differentialekvationer.

(Ekv. 2)

(Ekv. 3)

Oberoende av begynnelsevillkoren kommer populationen nå ett konstant slutvärde. Vi skulle därmed finna vilket konstant slutvärde för *V* och *S* dessa differentialekvationer uppnår om derivatan är 0.

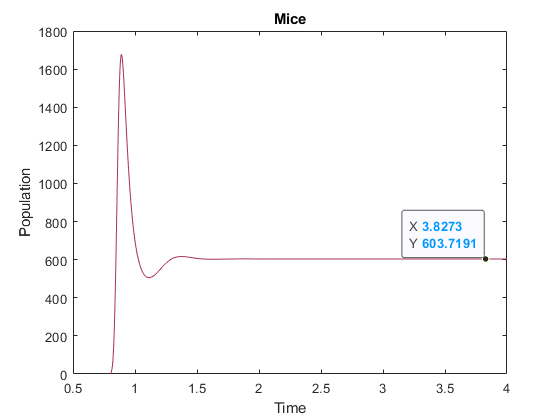
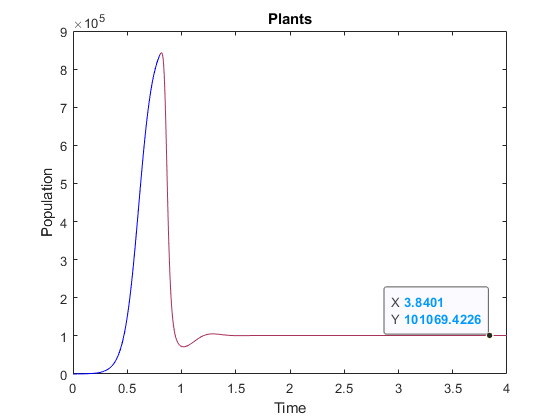
## **Metod**

Som tidigare nämnt hittas slutvärdena om derivatan är noll och vi fick därmed två olinjära ekvationer med två okända variabler. För att lösa detta ekvationssystem kunde vi använda Newtons metod för flera variabler som finns i *newtonsys.m*. För det hade vi vår vektorvärda funktion *fun* som bestod av Ekv. 2 och Ekv. 3 i första och andra komponenten. Dessutom hade vi vår jacobimatris som var följande:



Dessutom hade vi våra begynnelsevärden som var *V* = 100000 och *S* = 700*.* Vi hade testat oss fram till de begynnelsevillkoren. Vi tog även hjälp av en graf som ritade ut kurvan för nästa deluppgift för att bekräfta ifall resultatet från Newtons metod stämde. Vår iterativa metod fungerade med en while-loop som pågick tills felet, dvs. normen av *h*, var mindre än , vilket var vår tolerans. Loopen avbröts också ifall det har gått fler än 1000 iterationer.

## **Resultat**

De resultat vi fick från att använda Newtons metod för att hitta konstant slutvärdena var att *V* = 101069 och *S* = 603. Den numeriska approximationen som fick hade avrundades nedåt till närmaste heltal då man har hela växter och djur. För växterna är det en stor skillnad mellan detta svar och den population arten kunde uppnå när inga möss fanns. Osäkerheten i svaret undersöktes genom att köra samma kod fast med toleranser från till i funktionen *reliabilityNewton()*. Svaret ändrades inte vid mindre tolerans. Vi kan därmed säga att osäkerheten från denna metod ligger på den sista utskrivna decimalen som är avrundad. Däremot eftersom det finns hela djur och växter ligger osäkerheten på entalet om man tar hänsyn till verkligheten. *reliabilityNewton()* undersöker även ifall felen minskas kvadratisk mellan varje iteration, vilket den gjorde. 

**Figur 2 & 3:** Figurerna visar hur grafen går mot en konstant slutvärde när tiden ökar. Graferna skapades med Runge-Kutta 4 som används i “Växters och skadedjurs population mellan T1 och T2 =1.5”

Med hjälp av graferna i figur 2 och 3 verkade svaren vara inom en rimlig intervall för det verkade som att kurvan planade ut vid värden som stämde med resultatet från Newtons metod (se figur 2 och 3).

# Växters och skadedjurs population mellan T1 och T2 =1.5

## **Frågeställning**

Efter att vi har löst det olinjära ekvationssystemet skulle vi undersöka populationen vid T2 = 1.5 efter planteringen. Vi skulle undersöka ifall populationen hade stabiliserat sig och var nära slutvärdet samt hur mycket i procent/promille de avviker från slutvärdena.

## **Metod**

För att få ett approximativt värde på hur stor populationerna är från T1 till T2 = 1.5 användes Runge-Kutta 4 som finns i filen *rk4.m* och som använde funktionerna *rk4()* tillsammans med *calcRK()* för att lösa differentialekvationerna Ekv. 2 och Ekv 3.

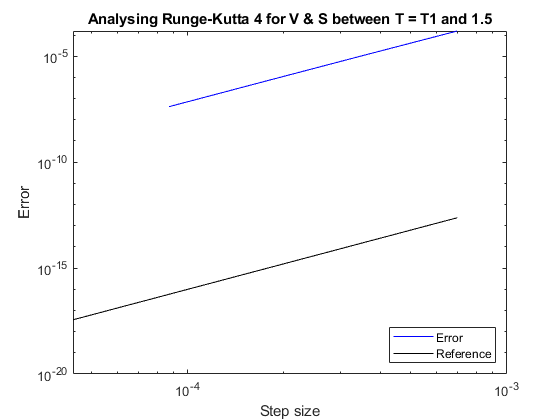
Till skillnad från när vi skulle hitta T1 med hjälp av Runge-Kutta och tidsbegränsningen var begränsad blev algoritmen för denna annorlunda när vi ska sluta vid T2 = 1.5. För denna gång var inmatningarna för *rk4()* att tidsbegränsningen *limit =* 1.5, *t0* = T1 och *s =* 2 då sluttiden var känd. Eftersom vi jobbade med flera variabler samtidigt innebar det att vi skulle lösa en första ordningens differentialekvationssystem. Den vektorvärda funktionen *fun* bestod av Ekv. 2 och Ekv. 3 som inmatning. Dessutom hade startvektorn *v* begynnelsevillkoret för *V* samt *S*, vilket var 95 % av slutvärdet för Ekv. 1 respektive *S =* 2 vid T1 eftersom det var givet i deluppgiften. Eftersom *s = 2* berodde steglängden *h* på antal delintervall, vilket var *N =* 1000 från T1 till T2 = 1.5. Därefter räknade algoritmen ut det approximativa värdet vid T2 med hjälp av Runge-Kutta 4 i *calcRK()* 1000 gånger och sparade tiden samt värdena efter varje iteration i en matris. Raderna i matrisen, uppifrån till ner, bestod av tiden, värdena för *V* samt värdena för *S.*

När vi fick fram värdena hade vi plottat alla datapunkter på en graf och analyserade ifall den planade ut vid T2. Dessutom räknade vi ut hur mycket de approximativa värdena skiljdes från konstant slutvärdena vi räknade fram tidigare. Skillnaden jämförde vi i promille.

## **Resultat**

De approximativa värdena, avrunda till närmaste heltal, som vi fick fram är ca *V =* 100660 och *S* = 607 som skiljer sig med 4.047 ‰ respektive 4.832 ‰ från konstant slutvärdena. Våra värden såg rimliga ut samtidigt som vi kunde ta hjälp av graferna från figur 2 och 3 som visade att vid T = 1.5 ska värdena vara runt där. Dessutom kunde vi avgöra från figur 2 och 3 att populationerna började stabilisera sig nära T = 1.5. Eftersom det stabiliserade sig där ska det vara nära konstant slutvärdena vi fick fram tidigare. Osäkerheten på det numeriska approximationen ligger på fjärde decimalen för *V* och på den sjätte decimalen för *S*. Däremot finns det hela växter och djur så då blir osäkerheten egentligen på entalet för båda. För att se osäkerheten hos promillen gjorde vi en experimentell störningsanalys när vi antog att från Newtons metod, *V* och *S* var 101069/101070 respektive 603/604. Från Runge-Kutta var *V* och *S* 100660/100661 samt 607/606. Det värde som inte hade störda inmatningar var promillen som är skriven ovan. Vi fick fram att *V* hade en osäkerhet på 0.0203 ‰-enheter och *S* hade 3.6005 ‰-enheter.

Vi undersökte osäkerheten i Runge-Kutta 4 metod på samma sätt vi gjorde i “T1 när växter uppnår 95 % av slutvärde”. Enda skillnaden var att istället för att direkt halvera steglängden, dubblerades antal delintervall som orsakade halvering i steglängden.



**Figur 4:** Figuren visar hur felen förhåller sig till steglängden och ifall den är parallell med referenslinjen som följer teorin för Runge-Kutta 4.

Återigen här med hjälp av figur 4 såg vi att linjerna var parallella med varandra, vilket betyder att Runge-Kutta 4 metoden fungerade som den ska enligt teorin (se figur 4). Dessutom undersökte vi att kvoten gick mot 16, vilket den gjorde. Sammanfattningsvis kunde vi säga att det finns en bra noggrannhet på approximationen.

# Växters, skadedjurs & rovdjurs konstant slutvärde

## **Frågeställning**

Efter T2 kommer den även två rovdjur *R*, vi säger ormar, var närvaro på ön påverkar samt påverkas av populationerna inom ekosystemet. Ekv. 2 för växterna är oförändrad men följande differentialekvationer modellerar för mössens och ormarnas population.

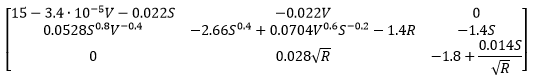
(Ekv. 4)

(Ekv. 5)

Det första vi skulle göra med Ekv. 2, Ekv 4 samt Ekv. 5 var kolla vilket konstantvärde populationerna går mot när tiden ökar.

## **Metod**

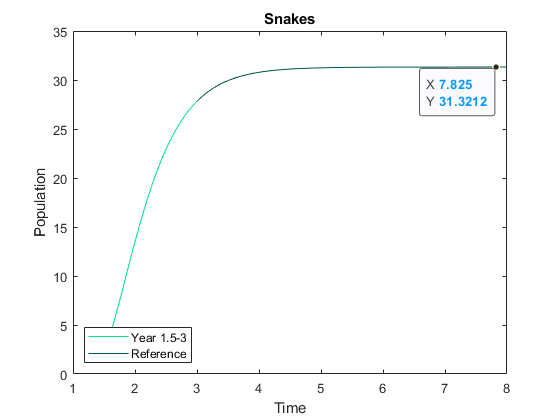
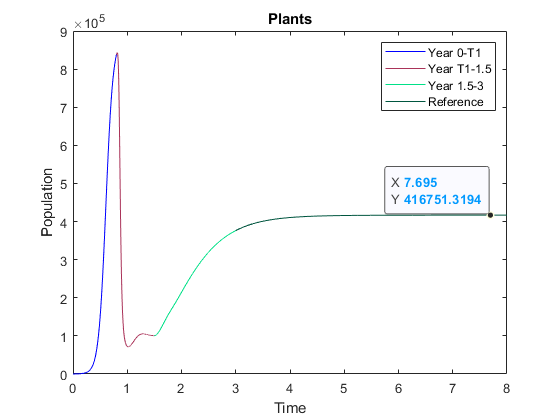
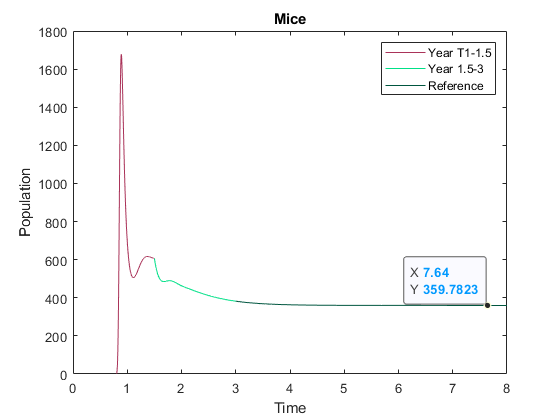
Mycket likt som vi gjorde i “Växters och skadedjurs konstant slutvärde” använde vi av oss Newtons metod för flera variabler och löste den numerisk med derivatan som noll. Vår vektorvärda funktion *fun* i *newtonsys()* innehöll Ekv. 2, Ekv 4 och Ekv. 5 i den ordningen. Vår jacobimatris var följande:



Våra begynnelsevärden var *V* = 100000, *S* = 600 & *R* = 20. Vi antog att slutvärdena för *V* och *S* inte skulle ändras signifikant och *R* antog vi skulle vara en storleksordning mindre än *S*. Dessa gav oss inga imaginära tal så dessa fungerade bra. Dessa hanteras av *newtonsys()* likadant som den gjorde med “Växters och skadedjurs konstant slutvärde” med samma tolerans med och while loop.

## **Resultat**

De konstanta slutvärdena populationerna når enligt Newtons metod var *V =* 416756, *S* = 359 och *R* = 31. Svaret är avrundade till lägsta heltal då populationen inte kan överstiga det värdet som ficks från metoden. Vi fann det rimligt att växter ökar när antalet möss minskar pga. ormars närvaro. När vi testade med olika toleranser med *reliabilityNewton()* fick vi exakt samma svar där alla har 14 decimaler noggrannhet. Siffrorna därefter är osäkra då den sista siffran avrundas. I verkligheten är dock osäkerheten på entalet då det kan finnas hela djur och växter. Felen i varje iteration minskade kvadratiskt som den ska enligt teorin i *reliabilityNewton()*.



**Figur 5, 6 & 7:** Figurerna visar hur grafen går mot en konstant slutvärde när tiden ökar. Graferna ficks fram med Runge-Kutta 4 som används i “Växters, skadedjurs & rovdjurs population tills T3 =3”

När vi använde Runge-Kutta 4 för att se vilket tal grafen gick mot ficks samma heltal som vi fick i vår numeriska approximation (se figur 5, 6 & 7). Därmed kunde vi dra slutsatsen att våra approximerade slutvärden log inom rätt intervall och var rimliga.

# Växters, skadedjurs & rovdjurs population tills T3 =3

## **Frågeställning**

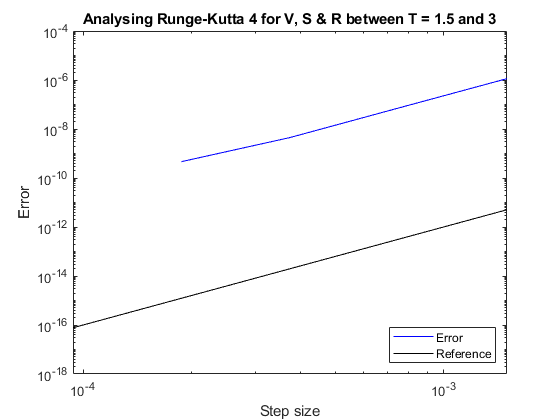
När slutvärdena har hittats ska differentialekvationerna Ekv. 2, Ekv. 4, Ekv. 5 lösas vid T3 = 3. Vi skulle främst undersöka hur nära populationerna hade kommit till sina slutvärden som ficks i “Växters, skadedjurs & rovdjurs konstant slutvärde”.

## **Metod**

Precis som med alla differentialekvationer och som vi gjorde i “Växters och skadedjurs population mellan T1 och T2 =1.5” löstes differentialekvationen numeriskt med Runga-Kutta 4 från T2 till T3 = 3. Vi använde samma kod och algoritm som finns i *rk4()* i filen *rk4.m*. 5 parametrar skickas till den funktionen från *PopdynMain.m*. Vi angav vår tidsbegränsning *limit* till 3, begynnelsetid *t0* till 1.5 och vår parameter *s* till 2 för att sluttiden var känd. Vår vektorvärda funktion *fun* innehöll, från första komponenten till tredje komponenten, differentialekvationerna Ekv. 2, Ekv. 4 och Ekv. 5. Vektorn *v0* med begynnelsevärdena innehöll populationen för *V* och *S* vid tiden T2 = 1.5. Den tredje komponenten hade *R* = 2 som var givet i deluppgiften. Delintervallerna är *N* = 1000 och steglängden anpassas efter antal delintervall och tidsintervallet.

## **Resultat**

De approximativa värden som ficks, avrundade till närmaste heltal, var *V =* 376380, *S =* 382 och *R* = 28. Det vi såg var att växternas population skiljde sig markant medan båda populationerna för mössen och ormarna var skillnaden inte signifikant. Med *reliabilityRK()* undersökte vi metodens noggrannhetsordning och svarens osäkerhet. Genom att jämföra värdena efter varje steglängdshalvering kommer vi fram till att *V* hade en osäkerhet i sjätte decimalen. *S* hade vid den nionde och *R* vid tionde decimalen. Dock eftersom det finns hela antal är osäkerheten på entalet.



**Figur 8:** Figuren visar hur felen förhåller sig till steglängden och ifall den är parallell med referenslinjen som följer teorin för Runge-Kutta 4.

Noggrannhetsordningen kanske inte var lika självklart. Kvoten gick mot 16 och sedan blev 9, vilket tydde på noggrannhet på 4 till en viss begränsning på steglängderna. Vi antog att detta berodde på trunkeringsfel vid mycket korta steglängd. Vi kollade med grafer med en referenslinje för fel som av avtar i fjärde ordningen. Endast i slutet såg vi att linjen inte var parallell med referenslinjen (se figur 8). Däremot eftersom vi använde större steglängd var vår metod tillräckligt noggrann då den delen av linjen är parallell med referenslinjen.

# Årlig besprutnings påverkan på populationerna

## **Frågeställning**

Vid tidpunkten T3 börjar en årlig besprutningskampanj vid slutet av året där 70 % av skadedjuren samt 20 % av rovdjuren dödas vid besprutning. Populationerna stabiliserar sig till nya värden och vi ska undersöka ifall öborna har gjort rätt.

## **Metod**

För att få fram vad som händer från T3 och 5 år framåt användes Runge-Kutta 4 som finns i filen *rk4.m* och använder funktionerna *rk4()* tillsammans med *calcRK()* som är algoritmen för självaste Runge-Kutta för differentialekvationer som Ekv. 2 och Ekv 3.

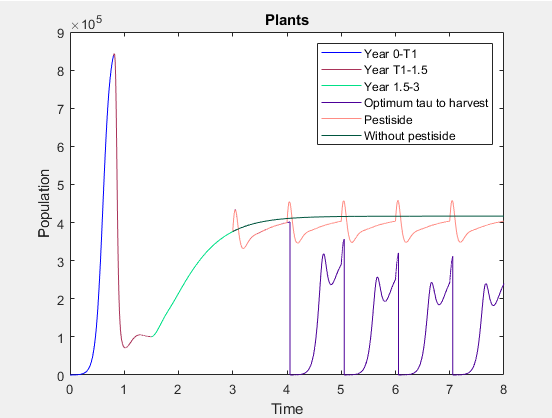
Till skillnad från när vi skulle hitta T2 & T3 med hjälp av Runge-Kutta och tidsbegränsningen var begränsad blev algoritmen för denna annorlunda när vi vid slutet på alla år måste minska beståndet av skadedjur till 30 % och rovdjur till 80 % (se bilaga 4). För denna gång var inmatningarna för *rk4()* att tidsbegränsningen;

*limit =* 3 + *i*, *t0* = T3 + *i* och *s =* 2 för att tidsspannet var känt. Vi gjorde ett år i taget eftersom vi i slutet av året satte och .

När vi fick fram värdena plottade vi alla värden tillsammans med en referenslinje som visade vad som skulle ha hänt om vi öborna inte börjar en årlig besprutningskampanj (se bilaga 5).

## **Resultat**

Öborna ville att skadedjuren skulle äta mindre av plantorna och därav få mer plantor.

****

**Figur 9:** Visar växtpopulationen över tid med/utan besprutning och med/utan skörd.

I grafen syns det att plant beståndet vid vissa delar på året var över referenslinjen och på visa delar av året var den inte. Vi kan också se att om vi skulle lägga in en normallinje för hur mycket växter som skulle finnas i snitt skulle den ligga under referenslinjen. Så beroende på vad öborna ville ha ut av besprutningskampanjen, om det var mer plantor i snitt eller mer plantor vid en viss tid, kan man dra olika slutsatser. Vi kan säga att besprutningskampanjen var gynnsam om och endast om en skörd införskaffas under den tiden på året då plant populationen var som högst. Men om man inte gör det och man skördade lite grann hela tiden skulle öborna att få mindre mat än före besprutningskampanjen.

# Modellens känslighet

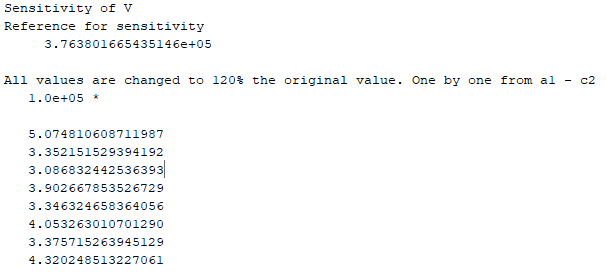
## **Frågeställning**

För nästa deluppgift skulle vi kolla hur känslig modellen är för störningar. För det skulle vi kolla vilken koefficient är V känsligast och detta kunde göras med ett numeriskt experiment.

## **Metod**

Precis som med alla differentialekvationer löstes detta numeriskt med Runga-Kutta 4. Vi använde samma kod och algoritm som finns i *rk4()* i filen *rk4.m*. Här skapades en ny funktion som heter *sensitivity* som tog två argument, båda var data för år 1 - 2. För att kunna avgöra vilken koefficient som var mest känslig till förändring ändrades alla koefficienterna en och en till 120 % sitt original värde. Sen kördes en *rk4()* fram till år 3, och därifrån sparades det sista värdet på V.

## **Resultat**

****

**Figur 10:** Visar vad som händer med V om vi ändrar koefficienternas värden

V var känsligast för koefficienten 15 (kallad a1 i uppgiften) (se figur 10). Detta var mest troligt ifrån att a1 har det största numeriska värdet och därför blev den numeriska ändringen i svaret som störst när vi satte den till 120 % sitt original värde. Felet i beräkningarna av V log i den sjätte decimalen alltså ±0.00001, då vi bara kan räkna på hela plantor.

# Tid för att skörda utan besprutning

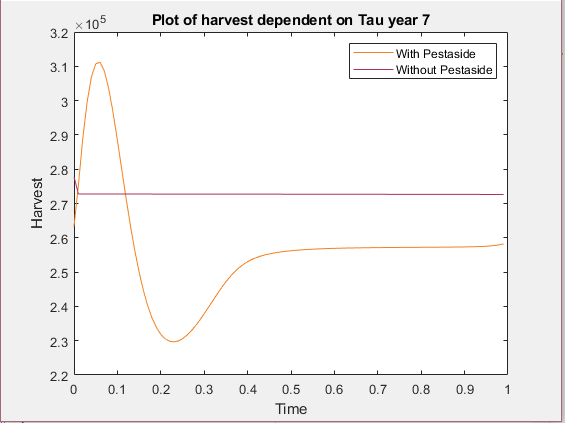
## **Frågeställning**

För utvidgningen för denna uppgift får vi det givet att öborna skördar från år 4 och sker vid *t =* 4 *+* τ, 5 *+* τ... , τ ∈ [0,1). De skördar allt förutom 100 växter. Vad vi skulle lösa differentialekvationssystemet med skörd dock utan besprutning. Detta ska lösas för τ = 0, 0.2, 0.5, 0.8 och dessutom ska vi plotta för *V, S* & *R.* Vi ska dessutom räkna hur stor skörden blir år 7 och undersöka ifall skörden beror på τ.

## **Metod**

För att hantera utvidgningens alla delar skapades en ny fil *expansionMain.m* och den kallades genom funktionen *expansionMain()* som tog ett argument *dataT3*. *expansionMain()* kallade i sin tur på en funktion *harvesting()* som tog argumenten; *dataT3* med datapunkter mellan T = 1.5 till 3*, tau, pest.* Här refererade *pest* till om vi ska ta hänsyn till besprutning eller inte (med siffran 1 eller 0, i detta fall 0). Funktionen byggde på “Årlig besprutnings påverkan på populationerna” fast den inkluderade en kontroll för skörd (se bilaga 6) Allt gick igenom en for-loop för att få de korrekta utskrifterna (se bilaga 7).

För att få en bild av hur skörden påverkades av τ användes en for-loop i funktionen *expansionMain()* (se bilaga 8). Den hämtade data för hur stor skörden blev år 7 beroende på τ*.* Sen plottades skörden mot τ*.*



**Figur 11:** Visar skörden med τ mellan 0 och 1, med och utan besprutning.

## **Resultat**

Från grafen såg vi att skörden inte varierade mycket när vi inte hade någon besprutning, skörden log alltid runt 2,7 \* 10^5 ± 0,1 (se figur 11).

För att säkerställa noggrannhetsordningen skapades *reliabilityExpansion.m.* Genom att jämföra värdena efter varje steglängdshalvering såg vi att noggrannhetsordningen blev 4 då kvoten blev 16.

Vi såg också med *reliabilityExpansion()* hur många decimaler i varje svar som var rätt, genom att jämföra värdena efter varje steghalvering;

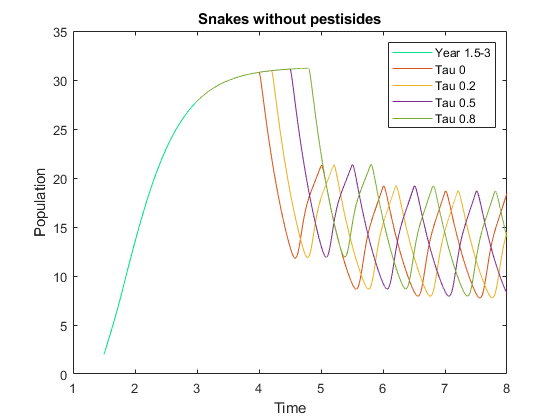
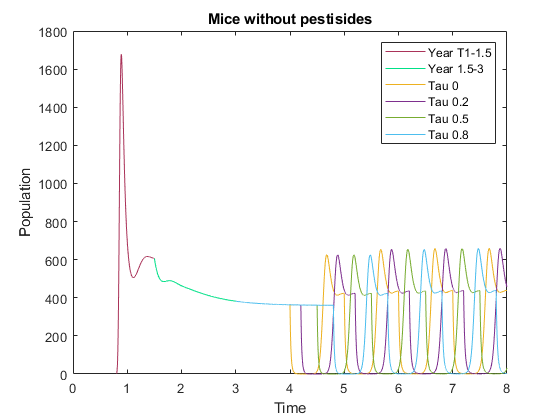
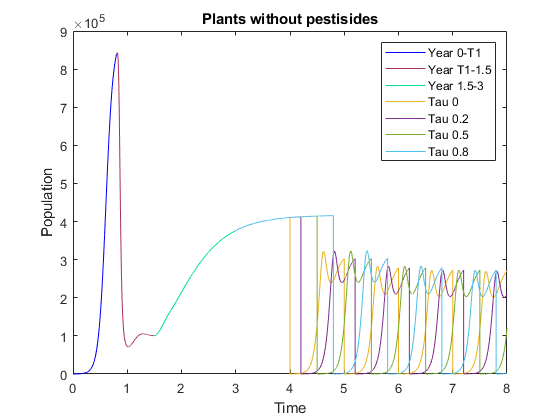
*tau = 0 -> 2.72* \*10^5 felet ligger i nästa decimal

*tau = 0.2 -> 2.58* \*10^5 felet ligger i nästa decimal

*tau = 0.5 -> 2.72* \*10^5 felet ligger i nästa decimal

*tau = 0.8 -> 2.72* \*10^5 felet ligger i nästa decimal

Plotten för V, S och R blir:



**Figur 12, 13 & 14:** Visar V, S och R för τ *=* 0, 0.2, 0.5 och 0.8 utan besprutning

# Tid för att skörda med besprutning

## **Frågeställning**

Denna gång skulle vi göra samma sak som vi gjorde i “Tid för att skörda utan besprutning”. Däremot denna gång skulle vi bekräfta att skörden berodde på τ.

## **Metod**

Som tidigare i “Tid för att skörda utan besprutning” användes samma funktioner (*expansionMain()* och *harvest()*). Men denna gång satts argumentet *pest* till 1 för att funktionen *harvest* skulle ta hänsyn till skörden som hände vid tiden *t +* τvarje år (se bilaga 6).

För att kolla om skörden berodde signifikant på τ prövades ett par olika *tau* (0, 0.2, 0.5, 0.8). Sedan plottades V, S och R för denna data (se bilaga 7). För att kolla hur skörden påverkades av besprutningen plottades skörden mot τ.

## **Resultat**

I figur 11 syns det att skörden nu starkt beror av τ, till skillnad från den tidigare uppgiften “Tid för att skörda utan besprutning”. Med *reliabilityExpansion()* fick vi fram hur stor skörden var samt hur många decimaler var rätt, genom att jämföra värdena efter varje steghalvering:

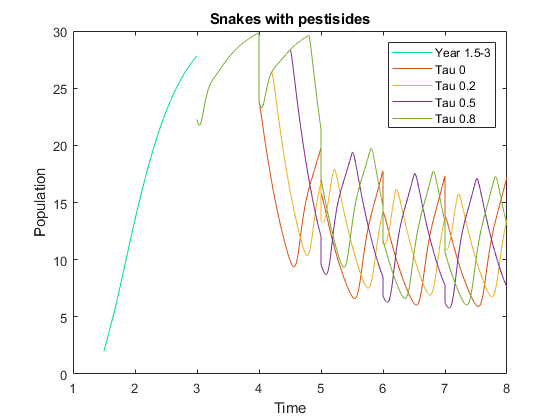
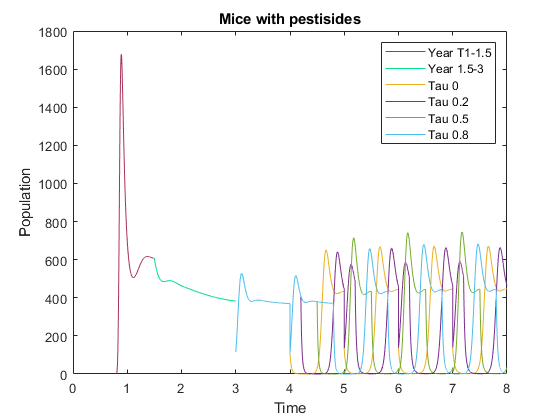
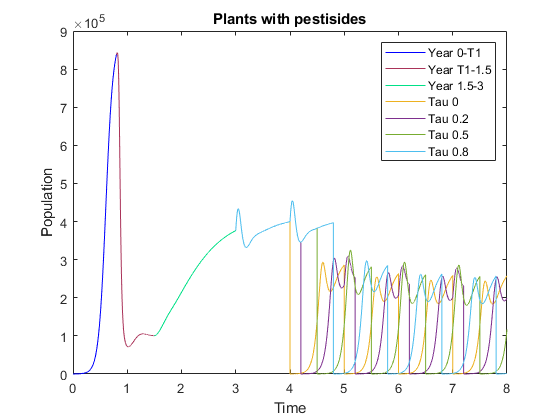
*tau = 0 ->* 2.585 \*10^5 felet ligger i nästa decimal

*tau = 0.2 ->* 1.99 \*10^5 felet ligger i nästa decimal

*tau = 0.5 ->* 1.16 \*10^5 felet ligger i nästa decimal

*tau = 0.8 ->* 1.44 \*10^5 felet ligger i nästa decimal

Plotten för V, S och R:



**Figur 15, 16 & 17:** Visar V, S och R för τ = 0, 0.2, 0.5 och 0.8 med besprutning

# Optimala tid för att skörda med besprutning

## **Frågeställning**

Den sista uppgiften var att hitta den optimala skördetiden när man använder besprutning. Vi skulle undersöka tillsammans med gyllene-snittet sökning vid vilken τ skörden var som störst år 7 samt ifall detta var större eller mindre än skörden utan besprutning. Dessutom skulle *V*, *S* & *R* plottas för det optimala τ-värde.

## **Metod**

För att kunna hitta den optimala skördetiden när man använder besprutning tog vi hjälp av beskrivningen som Gerd Eriksson har gett i sitt dokument *Numeriska algoritmer med Matlab.* Vi följde beskrivningen som var angiven där. Vår kod använde funktionen *harvest()* som de andra uppgifterna i utvidgningen. Argumentet *pest* var 1 i denna del, då besprutning ska vara med. Se bilaga 9. Hur många gånger while-loopen skulle köras anpassades till problemet. Här kördes while-loopen 100 gånger. Svaret skrevs ut i konsolen.

För att säkerställa antalet korrekta decimaler i utskriften kördes while-loopen, i bilaga 9, med *count < 10, count < 100, count < 1000*. Samt med *reliabilityExpansion()* får vi fram hur många decimaler som är rätt, genom att jämföra värdena efter varje steghalvering.

## **Resultat**

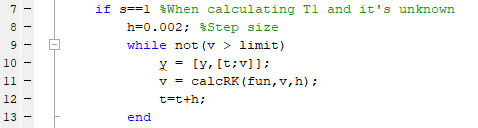
Vi såg direkt från figur 11 att skörden med besprutning gav en större skörd vid en viss tid på året. Med hjälp av gyllene snitt sökning kom vi fram till att den optimala tiden för skörd var τ *=* 0.0563. Osäkerheten log i nästa decimal.Oftast gör man dock en skörd över en dag och en dag är 1/365 = 0.00273972602. Vilket vid division med τ ger 20.5, vilket säger att det optimala tid för skörd är den 20 januari.

# Eget arbete

All kod och algoritmer har skrivits av oss själva. Grundläggande algoritmer på hur Runge-Kutta och Newtons metod ska skrivas har tagits inspiration från våra tidigare labbuppgifter. Algoritmen för hur man räknade optimala τ med gyllene snittet hade vi tagit hjälp av beskrivningen som Gerd Eriksson har gett i sitt dokument *Numeriska algoritmer med Matlab.* Vi följde endast beskrivningen som var angivet där och har skrivit koden själva. Vid slutet av projektet har vi dubbelkollat metoderna och resultat med två andra grupper. Vi har inte delat kod under detta tillfälle utan redovisat och motiverat för varandra hur vi har tänkt. Vi har hjälpt andra grupper med hur man ska tänka och lösa uppgifterna. Här har inga kod delats.

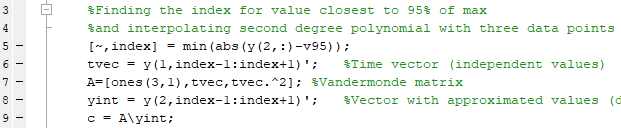
# Bilagor

**Bilaga 1:**



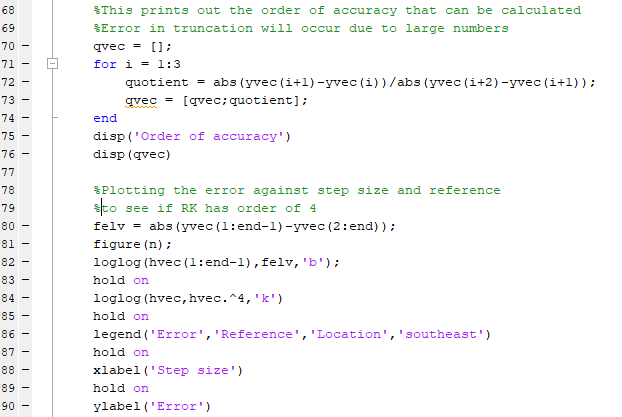
[Bilagan visar från *rk4()* för när alla datapunkter räknas ut när T1 sökes]

**Bilaga 2:**



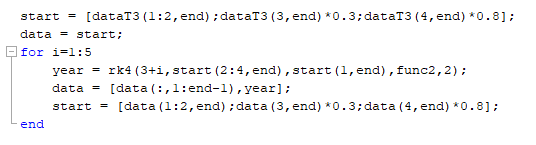
[Bilagan visar hur interpolationen genomfördes och hur relevanta vektorer samt Vanadermodematrisen såg ut.]

**Bilaga 3:**



[Bilagan visar ett kodstycke från *reliabilityRK()* för hur felet plottas på en graf mot steglängden tillsammans med en referenslinje.]

**Bilaga 4:**



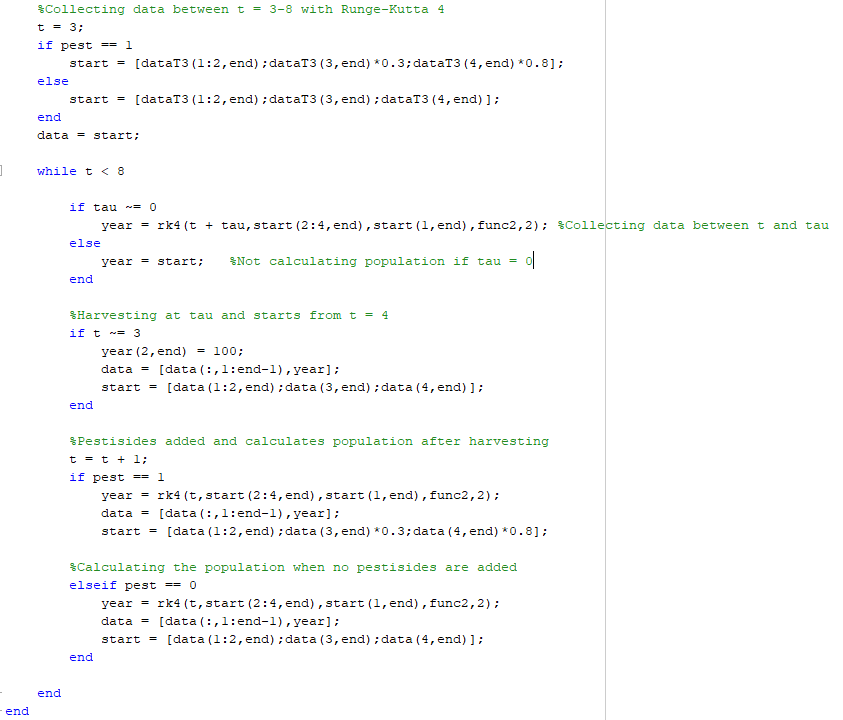
[Bilagan visar koden för besprutningskampanjen.]

**Bilaga 5:**



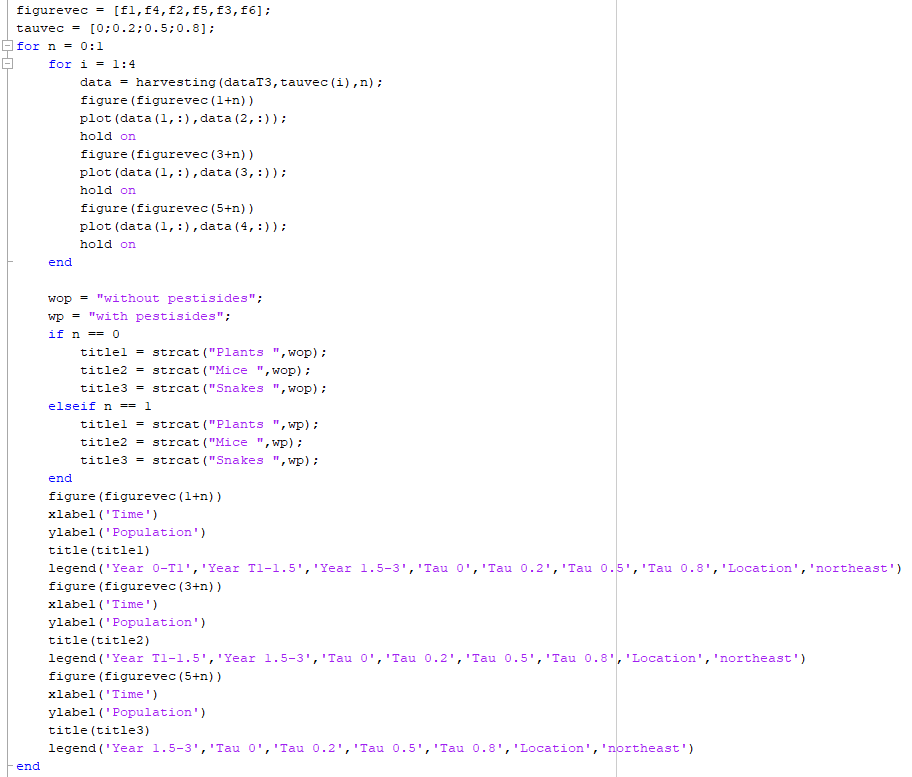
[Bilagan visar referenslinjen för besprutningskampanjen.]

**Bilaga 6:**



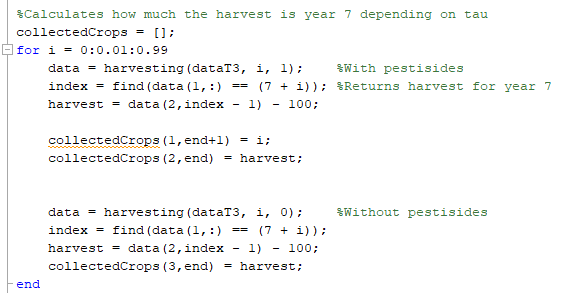
[Bilagan visar kodstycke från *harvesting* i filen *expansionMain.m*]

**Bilaga 7:**



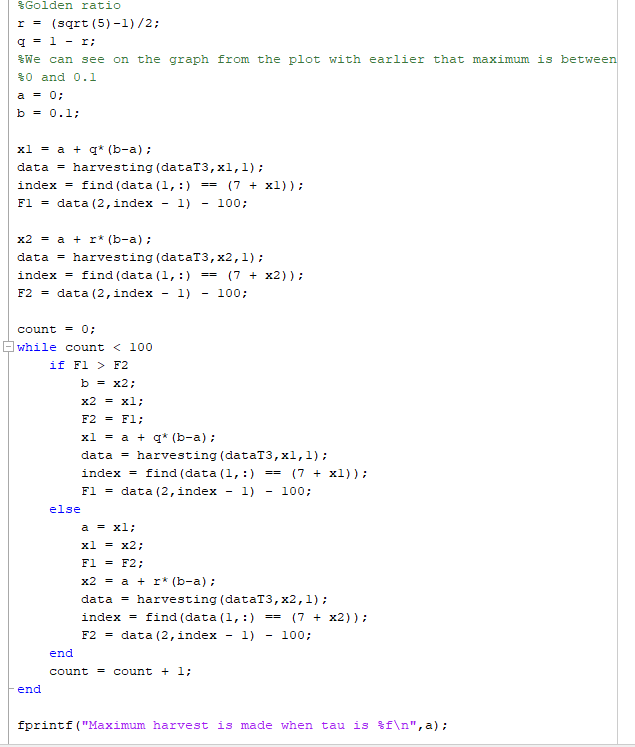
**[**Bilagan visar hur utskriften från *expansionMain* ser ut. Den skriver ut får både med och utan besprutning.]

**Bilaga 8:**

****

**[**Bilagan visar kodstycke från funktionen *expansionMain* som hämtar information om effekten *tau* har på skörden.]

**Bilaga 9:**



[Bilagan visar kod adaptionen av beskrivningen för gyllene snitt sökning från boken *Numeriska algoritmer med Matlab av* Gerd Eriksson.]