Projekt - Eksperimentiel fysik og statistisk dataanalyse

Lanchester modellen for krigsførsel

Freja Maria Dithmer Gam (studienummer: 202007434)
& Peter Asp Hansen (studienummer: 202005088)

19. maj 2021

 [1]

[1] http://www.battleofkursk.org/Battle-of-Kursk-Tanks.html

1 Abstrakt

Dette projekt omhandler analysering af krige vha. Lanchesters kvadratlov, mere specifikt Slaget ved Kursk. Iminuit pakken anvendes til at fitte data fra Slaget ved Kursk til løsningen til de koblede differentialligninger som Lanchesters kvadratlov består af. Der laves Monte Carlo simuleringer af løsningerne, og det undersøges hvor det er mest sandsynligt at løsningen til de koblede differentialligninger befinder sig. Derudover laves der en χ^2 hypotesetest, som bruges til at give en indikation af hvor effektiv Lanchesters kvadratlov er til at simulere forløbet af et krigsslag. Herefter diskuteres begrænsningerne i brugen af Lanchesters kvadratlov til analyse af krig, og afslutningsvist konkluderes det at Lanchesters kvadratlov rimelig effektivt simulerer Slaget ved Kursk på grund af slagets simple natur, såfremt dag 1 og dag 2 er ekskluderet fra analysen pga. minefelter som der ikke kan tages hensyn til med Lanchesters kvadratlov.

Indholdsfortegnelse

1 Abstrakt **2**

2 Indledning **4**

3 Teori **4**

3.1 Lanchester modellen 4

3.2 Partielle løsning 5

4 Historisk baggrund for slaget ved Kursk **6**

5 Datanalyse **7**

5.1 Iminuit 7

5.2 Monte Carlo 9

5.3 $\boldsymbol{\chi^2}$ hypotesetest 11

6 Diskussion **13**

7 Konklusion **14**

8 Bilag **15**

8.1 Original data 15

8.2 Kode 16

8.2.1 Iminuit 16

8.2.2 Monte Carlo 18

8.2.3 χ^2 hypotesetest 19

2 Indledning

Militærkræfter har i lang tid inkorporeret matematiske modeller for at forøge deres effektivitet i krig. En måde hvorved man kan inkorporere matematiske modeller er vha. koplede differentialligninger også kaldt for Lanchesters modeller, navngivet efter den Britiske matematiker Frederick W. Lanchester. I virkeligheden har han udviklet flere koblede differentialligninger til at modellere forskellige former for slag, men i dette projekt vil vi koncentrere os om Lanchesters kvadratiske lov, beskrevet i sektionen nedenfor, som bruges til at beskrive konventionelle kampslag. Ved særlige forhold vil denne model kunne bruges til at forudsige udfaldet af en kamp og analysere årsager til et udfald. Det kan virke meget ambitiøst at påstå, at et slag kan blive beskrevet af en matematisk model, eftersom det kan virke meget uforudsigeligt med et utal af variabler som man skal tage højde for. Derfor er det logisk at spørge sig selv, hvor effektiv er Lanchester kvadratiske lov til at simulere forløbet af et krigslag? Formålet med projektet er derfor at evaluere Lanchesters kvadratiske lov ud fra en statistisk dataanalyse af historisk data fra slaget ved Kursk.

3 Teori

Nedenfor præsenterer vi Lanchesters kvadratiske lov samt den tilsvarende teoretiske baggrund.

3.1 Lanchester modellen

Lanchester kvadratlov er et sæt koblede differentialligninger, som beskriver krigsførsel mellem to kampstyrker.

$$\frac{dG}{dt} = -rR(t)$$
$$\frac{dR}{dt} = -gG(t)$$

Der findes flere modeller til at analysere krigsførsel, men vi har valgt kun at udforske denne. Lanchester kvadratiske lov beskriver tidsafhængigheden af styrken af to rivaliserende tropper. I ligningerne repræsenterer R(t) og G(t) antallet af russiske og tyske tropper, mens r og g repræsenterer deres respektive kampstyrke. Således er raten hvorved f.eks. tyske tropper bliver elimineret

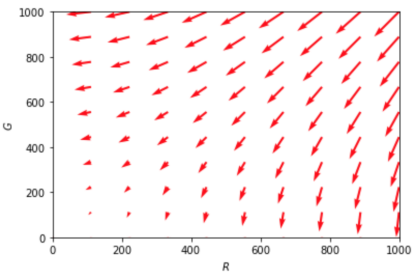

Figure 1: Faseplot

proportionel med antallet af russiske tropper.
(Baktoft, A, 2017, s. 131)

Differentialligningerne kan benyttes til at skabe et faseplot som vi senere hen kommer til at sammenligne med virkelig data. Et sådanne faseplot kan i vores tilfælde se ud som i figur 1. I et hvert slag, med mindre en af fronterne overgiver sig, vil udfaldet blive besluttet ud fra hvilken side der først rammer et antal tropper på nul. Det er dermed logisk at denne side har tabt slaget. Udfaldet kan derfor besluttes kvalitativt ud fra faseplottet. Det mistænkes ud fra denne figur at den afledte funktion til faseplottet er en hyperbolsk funktion. Hvis denne funktion derfor krydser "R-aksen", har de russiske tropper vundet slaget, mens krydser funktionen "G-aksen" har de tyske tropper vundet slaget. I den næste sektion vil vi påvise at den partielle løsning til de koblede differentialligninger er en hyperbolsk funktion samt give et bud på hvordan man kan bestemme udfaldet af et slag kvantitativt.

3.2 Partielle løsning

Den bedste måde at beskrive udviklingen af et slag ud fra Lanchesters kvadratisk lov, er ved at finde løsningen til de koblede differentialligninger. Vi kan allerede nu bestemme den partielle løsning, mens vi senere kommer til at bestemme den fulde løsning numerisk. For at bestemme den partielle løsning indleder vi med at dividere de to differentialligninger med hinanden.

$$\frac{\frac{dG}{dt}}{\frac{dR}{dt}} = \frac{dG}{dR} = \frac{-rR(t)}{-gG(t)}$$

Således har vi en første-grads differentialligning som vi kan løse vha. separation af variable.

$$\int -gG(t)dG = \int -rR(t)dR$$
$$-\frac{g}{2}G(t)^2 + c_1 = -\frac{r}{2}R(t)^2 + c_2$$

Vi får dermed at den partielle løsning til de koblede differentialligninger er en hyperbolsk funktion som beskrevet på følgende form:

$$\frac{R(t)^2}{c/r} - \frac{G(t)^2}{c/g} = 1$$
$$rR(t)^2 - gG(t)^2 = c$$

Ud fra den hyperbolske funktion kan vi bestemme udfaldet kvalitativt ud fra værdien af c. Er værdien af c f.eks. positiv indikere det at $rR(t)^2 > gG(t)^2$, og dermed vinder de russiske tropper. Det modsatte udfald vil opstå hvis c er negativ. (MacKay, 2005)

Vi vil nu opstille den hypotese at slaget ved Kursk kan blive modelleret af den hyperbolske funktion udledt fra de koblede differentialligninger i sektion 3.1. For yderligere at bevise at

Lanchesters kvadratiske lov er egnet til at modellere dette slag er vi nødsaget til at betragte det historiske perspektiv.

4 Historisk baggrund for slaget ved Kursk

Slaget ved Kursk var et slag under 2. verdenskrig på Østfronten, mellem tyske og sovjetiske styrker. Det begyndte d. 4 juli 1943, og varede 15 dage før en sovjetisk sejr. De primære styrker i slaget var i form af kampvogne, og på trods af at der også har været andre slags styrker, antager vi at udfaldet af slaget udelukkende afhænger af kampvogne og deres tilhørende kampstyrke. I begyndelsen stødte de tyske tropper på minefelter, som forårsagede et stort tab af tyske tropper i de første par dage. De første 8 dage af slaget var tyskerne offensive, hvorefter de sovjetiske tropper angreb de resterende 7 dage, og endte med at vinde slaget. I nedenstående tabel kan data fra slaget ses. De originale data kan ses under bilag, sektion 8.1. I de nedenstående data er der taget højde for de tanks der er blevet tilføjet til tropperne undervejs, hvilket der ikke er i de originale data. Eftersom de førnævnte minefelter ville være svært at tage højde for i brug af Lanchester modellen, blev de to første dages data ekskluderet fra dataanalysen. Dette valg bliver diskuteret yderligere i afsnit 6.

Antal tanks i slaget ved Kursk				
Dag	R	ΔR	G	ΔG
1	2500	0	1178	4
2	2395	105	980	198
3	2278	117	732	248
4	2019	259	611	121
5	1704	315	503	108
6	1415	289	364	139
7	1258	157	328	36
8	1123	135	265	63
9	709	414	167	98
10	592	117	110	57
11	474	118	64	46
12	378	96	-15	79
13	351	27	-38	23
14	309	42	-45	7
15	224	85	-51	6

Table 1: Historisk data

(Turkes, 2000, s. 29, 31) Vi har allerde fra tidligere defineret R og G som værende henholdsvis antallet af russiske og tyske tropper. Derudover angives også faldet i antallet af tropper ΔR og ΔG fra dag til dag.

5 Datanalyse

Det er nu vores intention at fitte den historiske data til den model vi præsenterede i sektion 3.

5.1 Iminuit

Vi kan bestemme kampstyrken for de Russiske og Tyske tropper vha. pakken Iminuit. Denne pakke ligner på mange måder curvefit pakken med den forskel at man i denne pakke selv kan definere vores goodness of fit parameter. Dette er praktisk eftersom vi derfor kan antage at dataen er poisson fordelt og vi derfor kan definere vores goodness of fit parameter:

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}$$

Det giver logisk mening at dataen er poisson fordelt eftersom antallet af russiske og tyske tanks er tælleværdier. Hvorvidt dataen er poisson fordelt afhænger derudover også af en række andre kriterier. Heriblandt kræves det at alle begivenheder er uafhængige. Dette er ikke tilfældet, derfor at det vigtigt at pointere, at vi antager at dataen er poisson-fordelt på trods af dette. Således bliver vores goodness of fit parameter:

$$\chi^2 = \sum_i \frac{(G_{O,i} - G_{E,i})^2}{G_{E,i}} + \sum_i \frac{(R_{O,i} - R_{E,i})^2}{R_{E,i}}$$

Hvor G_0 og R_0 er det observerede antal af tyske og russiske tropper, mens G_E og R_E er det forventede antal af tyske og russiske tropper.

Vi kan således nu bestemme kampstyrkerne som fitte-parametrene g og r. Iminuit-pakken tager tre inputs, som er henholdsvis goodness of fit parameteren samt de respektive gæt for g og r.

```
m = Minuit(GoF, g=0.1, r=0.03)
m.errordef = 1
m.migrad()
m.hesse()
```

(Ongmongkolkul, 2020)

Goodness of fit parameteren definerer vi på følgende måde:

```
def GoF(g, r):
    res = solve_ivp(dydt, [t[0], t[-1]], [G[0], R[0]], args=(g, r),
        t_eval=np.linspace(t[0], t[-1], len(t)))
    G_E = res.y[0]
    R_E = res.y[1]
    z1 = (G-GE)**2/GE
    z2 = (R-RE)**2/RE
    z = np.append(z1,z2)
    return np.sum(z)
```

Her bestemmes de forventede værdier G_E og R_E vha. SciPy, hvor vi løser differentialligningen dydt som er defineret ud fra de koblede differentialligninger.

```
def dydt(t, y, g, r):
    return [-r*y[1], -g*y[0]]
```

Således får vi det følgende output:

	Name	Value	Hesse Err
0	g	0.614	0.006
1	r	64.3e-3	0.6e-3

Table 2: Output fra Iminuit

Herfra kan vi aflæse fitteparameterne:

$$g = 0.614 \pm 0.006$$

$$r = 0.0643 \pm 0.0006$$

Vi kan således allerede nu bestemme en af løsningerne til de koblede differentialligninger vha. den hyperbolske funktion fra sektion 3.2:

$$\frac{R(t)^2}{c/r} - \frac{G(t)^2}{c/g} = 1$$

Indsætter vi startværdierne for R(t) og G(t) kan vi bestemme c.

$$\begin{aligned} c &= rR(t)^2 - gG(t)^2 \\ &= R(t)^2 \cdot 0.0643 - G(t)^2 \cdot 0.614 \\ &= 2278^2 \cdot 0.0643 - 732^2 \cdot 0.614 \\ &= 4675 \end{aligned}$$

Vi kan dermed se ud fra værdien af c, at Lanchester modellen forudser russerne som vinder af slaget ved Kursk, eftersom c er positiv, altså er $rR(t)^2 > gG(t)^2$. Dette stemmer overens med det faktiske udfald af slaget. Ligeledes får vi følgende hyperbolske funktion som vores fittefunktion:

$$\frac{R(t)^2}{4675/0.0643} - \frac{G(t)^2}{4675/0.614} = 1$$
$$\frac{R(t)^2}{72706} - \frac{G(t)^2}{7614} = 1$$

Ligeledes kan vi plotte den historiske data sammen med modellen.

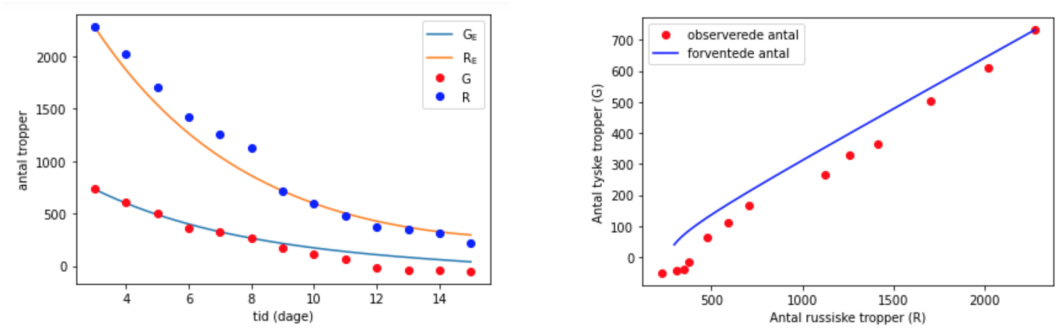

Figure 2: Tids- og faseplot af tyske og russiske tropper

5.2 Monte Carlo

Vi kan nu vha. Monte Carlo simuleringer, simulere alle de mulige løsninger til de koblede differentialligninger. Monte Carlo er en numerisk metode til at udregne mængder såsom integraler, sandsynligheder og konfidensintervaller. I dette tilfælde, med koblede differentialligninger, kan man ikke finde et analytisk udtryk for usikkerhederne, og derfor laves der en Monte Carlo simulering til at give et billede af usikkerhedernes betydning. Vi Monte Carlo simulerere dermed resultatet fra forrige sektion:

$$g = 0.614 \pm 0.006$$
$$r = 0.0643 \pm 0.0006$$

Vi definere således middelværdien og spredningen af dataen som:

	g	r
mean (μ)	0.614	0.0643
std (σ)	0.006	0.0006

Table 3: Fitteparametre

Vha. Monte Carlo kan vi simulere nye værdier for g og r, "nrep" gange ud fra en normalfordeling. Simuleres der 100000 gange får vi følgende fordelinger for g og r:

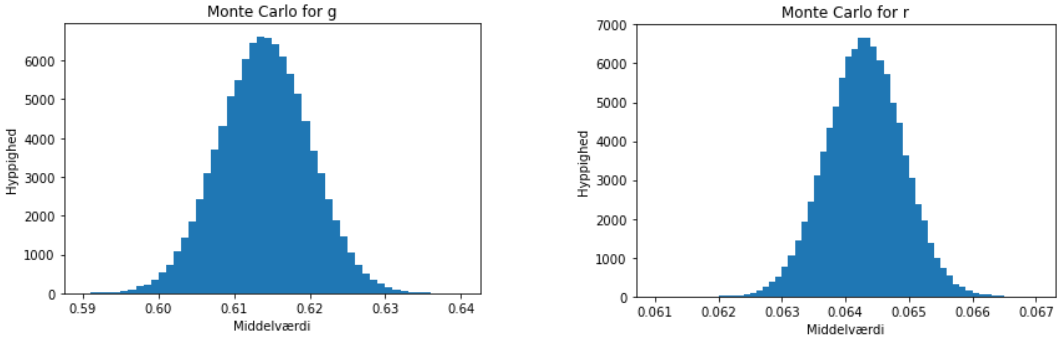

Figure 3: Spredninger for g og r

Således kan vi nu plotte alle de mulige løsningen til differentialligningerne:

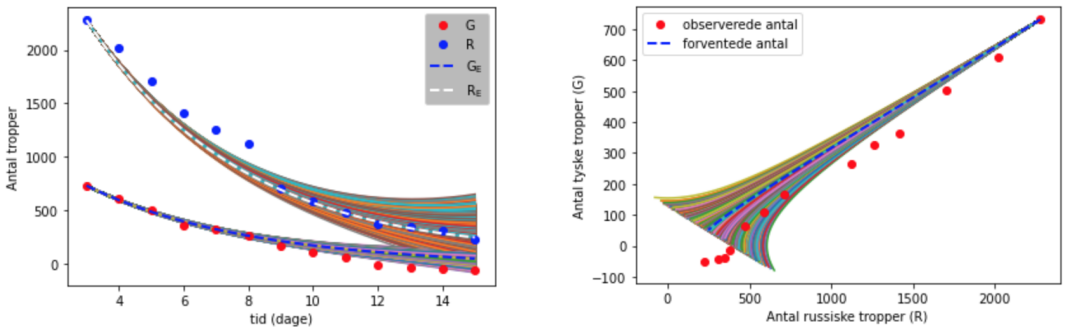

Figure 4: Løsninger til de koblede differentialligninger

De stiplede linjer er blot én ud af de 100000 plottede løsninger. Det er bemærkelsesværdigt at indenfor den givne usikkerhed, er det også muligt for tyskerne at vinde slaget.

Vi plotter nu følgende par af værdier for g og r, for at give en bedre indikation om hvor det er mest sandsynligt at løsningen til de koblede differentialligninger befinder sig.

#	g	r
1	g	r
2	$g - \sigma_g$	$r - \sigma_r$
3	$g + \sigma_g$	$r + \sigma_r$
4	$g - \sigma_g$	$r + \sigma_r$
5	$g + \sigma_g$	$r - \sigma_r$

Table 4: talpar

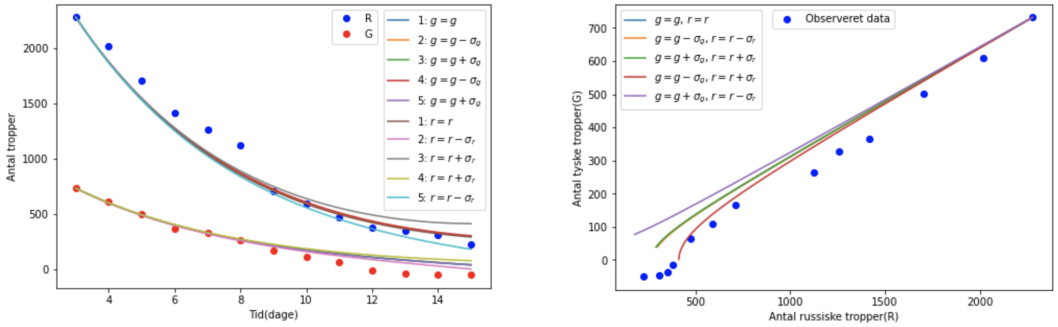

Figure 5: Mest sandsynlige løsninger

Sammenligner vi figur 5 med figur 4, anses det at der ikke er helt så stor variation i de forskellige løsninger, mens at udfaldet stadig kan svinge til begge sider.

5.3 χ^2 hypotesetest

Vi udfører nu en χ^2 hypotesetest til at bestemme kvaliteten af vores fit for faseplottet. Vi vil dermed bestemme p værdien ved først at liniarisere den hyperbolske funktion.

$$\frac{R(t)^2}{72706} - \frac{G(t)^2}{7614} = 1$$

Vi isolerer dermed G(t).

$$G(t) = \sqrt{7614} \cdot \sqrt{\frac{R(t)^2}{72706} - 1}$$

Vi definere således et nyt variabel x som:

$$x = \sqrt{\frac{R(t)^2}{72706} - 1}$$

Således at vi har følgende lineære funktion:

$$G(t) = \sqrt{7614} \cdot x$$

Vi vil således lave en χ^2 hypotesetest af x. Hertil opstilles der nu en tabel over de observerede værdier for x samt de forventede værdier for x.

tid (dag)	Observerede data / x_O	Forventede data / x_E
3	8.39	8.39
4	7.42	6.86
5	6.24	5.61
6	5.15	4.58
7	4.56	3.73
8	4.04	3.06
9	2.43	2.46
10	1.95	1.97
11	1.45	1.57
12	0.98	1.23
13	0.83	0.94
14	0.56	0.68
15	nan	0.45

Table 5: Liniarisering

Ved at plotte denne data kan vi visualisere liniariseringen.

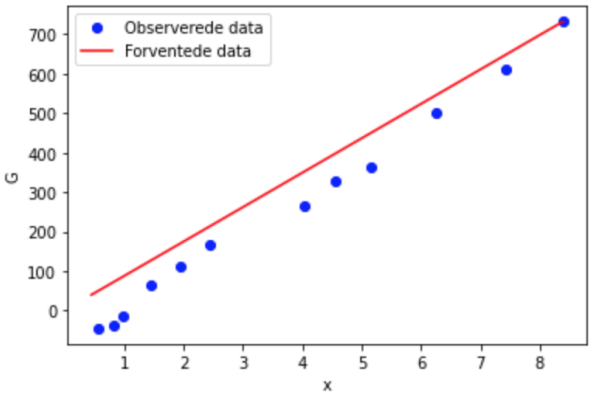

Figure 6: Liniarisering

Vi er nu parat til at foretage den fulde χ^2-test. Vi bestemmer først en nul-hypotese:

H_0: Slaget ved Kurske kan modelleres ud fra Lanchesters kvadratlov.

Denne hypotese bliver påvist hvis p-værdien er indenfor konfidensintervallet.

Således kan vi også bestemme den alternative hypotese:

H_1: Lanchesters kvadratlov er ikke effektivt anvendelig til at modellere slaget ved Kursk.

Denne hypotese bliver påvist hvis p-værdien er udenfor konfidensintervallet.

Signifikans-niveauet sættes til 5% og antallet af frihedsgrader bestemmes vha. følgende formel:

$$\nu = N - 1$$

Hvor N repræsentere antallet af datapunkter. Derudover trækkes antallet af variable også fra i vores tilfælde er det 1 for x.

```
chmin=np.sum((x0[0:12]-xE[0:12])**2/xE[0:12])
p = ss.chi2.sf(chmin,len(x0)-1)
print(chmin)
```

Således får vi et output der giver os χ^2-teststørrelsen samt p-værdien:

$$\chi^2 = \sum_{i=1}^{N} \frac{(O_i - E_i)^2}{E_i} = \sum_{i=1}^{N} \frac{(x_0 - x_E)^2}{x_E} = 0.7946854523507199$$

$$P(\chi^2_{min}; \nu) = 1 - chi2.cdf(\chi^2, \nu) = 0.9999844839160952$$

Denne værdi for p er indenfor konfidensintervallet eftersom $P > 0.05$. Dermed ved første glans anses nulhypotesen for at være rigtig. Der gælder dog at når $P(\chi^2_{min}; \nu) \to 1$ stilles der spørgsmåltegn til usikkerhederne af dataen. (Hughes, I. G, T. P. A. H, 2010, s. 106) Dette skyldes sandsynligvis de tidligere komplikationer vi havde, hvor vi antog at dataen er Poisson fordelt selvom den i virkeligheden ikke er det.

6 Diskussion

En simpel krig som Slaget ved Kursk er mulig at analysere vha. Lanchesters kvadratlov og komme frem til en rimeligt resultat. Denne metode er dog i sig selv meget simpel, og kigger udelukkende på hvor mange tropper der er til at starte med, hvor mange der dør undervejs og deres kampstyrke. I virkeligheden er krige noget mere komplekse, især i nyere tid, hvor mange nye og uforudsigelige midler er taget i brug. I hvor høj grad man kan bruge Lanchesters kvadratlov til at analysere krige - nyere eller gamle - er derfor begrænset.

I analysen af Slaget ved Kursk antager vi f.eks. at data er poisson fordelt, på trods af at begivenhederne tydeligvis er afhængige af hinanden, hvilket egentligt går direkte imod et af kravene for at data er poisson fordelt. Undervejs i slaget, tilføjede tyskerne nogle tanks til deres tropper, hvilket der i den originale data ikke tages højde for, men dette tog vi højde for i de data vi har analyseret på - dog resulterer dette i at noget af dataen bliver negative tal. Lige i dette slag er det til at manipulere data så det er muligt at analysere på, men ved større

og mere komplicerede krige, kunne det blive problematisk at holde styr på hvilke styrker der bliver tilføjet undervejs.

En anden begrænsning i at bruge Lanchesters kvadratlov på Slaget ved Kursk er at den antager at kampstyrken forbliver konstant under hele slaget, hvilket den ikke nødvendigvis gør. F.eks. kan tanks blive beskadiget uden at trække sig fra slaget, hvilket ville formindske den reelle kampstyrke betydeligt. I de fleste krige er der desuden forskellige typer af tropper; der kunne være tanks, fly, kemiske våben, etc., som alle har forskellige styrker og svagheder. Lige netop i Slaget ved Kursk, bliver dette ikke et problem, da der udelukkende blev brugt tanks, som derfor er sammenlignelige, men i en knapt så konventionel krig, med mange styrker og forskellige typer af våben og tropper, ville det være meget besværligt - hvis ikke umuligt - at modellere.

I denne opgave er data fra dag 1 og dag 2 ekskluderet fra dataanalysen, pga. de minefelter tyskerne stødte på i disse dage. Dette blev, som tidligere nævnt, gjort for at undgå at fitte til noget som vi på forhånd godt vidste at Lanchester modellen ikke kunne forholde sig til, og hvis disse punkter blev medtaget i dataanalysen ville de forårsage at fittet passede dårligt til alle de andre punkter. For at Lanchesters kvadratiske lov nemlig skal virke, kræves det også at hver kampenhed maksimalt kan eliminere én af modstanderens kampenheder af gangen. Derfor anses de koblede differentialligninger at være uegnet til at modellere slag hvor maskingeværer, artilleri, atomvåben, miner og andre masseødelæggelsesvåben indgår. (Baktoft, A, 2017, s. 129-139)

7 Konklusion

Ud fra dataanalysen, kan det konkluderes at Lanchesters kvadratlov er i stand til at rimelig effektivt simulere Slaget ved Kursk, såfremt dag 1 og 2 er ekskluderet i analysen. Slaget ved Kursk blev valgt til dette projekt, på grund af de begrænsede faktorer der kunne påvirke effektiviteten af Lanchesters kvadratlov, og det faktum at der selv i dette tilfælde er flere ting som Lanchesters kvadratlov ikke kan tage hensyn til, viser de mange begrænsninger der er ved brug af denne lov, som kan gøre det svært at forudsige udfaldet af en krig, og hvad der forårsagede det. Desværre var der begrænsede data fra andre konventielle krige, som kunne have været interessante at anvende Lanchesters kvadratlov på, og sandsynligvis givet en bedre forståelse for betydningen af begrænsningerne ved loven. Lanchesters kvadratlov er siden dens opfindelse blevet modificeret til at kunne overkomme nogle af disse begrænsninger, og håndtere analyse af andre slags krigsførelse, f.eks. Guerilla krigsførelse, og det ville desuden være interessant at undersøge nogle krige vha. disse modificerede love.

Bibliografi

Baktoft, A (2017), *Matematik i virkeligheden*, 1. edn, Natskyggen.

Hughes, I. G, T. P. A. H (2010), *Measurements and their uncertainties*, Oxford University Press.

MacKay, N. (2005), 'Lanchester combat models', https://arxiv.org/pdf/math/0606300.pdf. Hentet 12. maj 2021.

Ongmongkolkul, P. (2020), 'Iminuit tutorial', https://nbviewer.jupyter.org/github/scikit-hep/iminuit/blob/master/tutorial/basic$_t utorial.ipynb. Hentet 12. maj 2021$.

Turkes, T. (2000), 'Fitting lanchester and other equations to the battle of kursk data', http://hdl.handle.net/10945/7700. Hentet 12. maj 2021.

8 Bilag

8.1 Original data

Day	OH Tanks	Damaged	Dst+Abnd	Total Loss
1	1178	2	2	4
2	986	175	23	198
3	749	216	32	248
4	673	107	14	121
5	596	92	16	108
6	490	107	32	139
7	548	32	4	36
8	563	48	15	63
9	500	89	9	98
10	495	50	7	57
11	480	32	14	46
12	426	70	9	79
13	495	15	8	23
14	557	6	1	7
15	588	6	0	6
TOTAL		**1047**	**186**	**1233**

Table 11. German tank data. Dst+Abnd denotes destroyed and abandoned tanks. OH denotes the on hand amount.

Figure 7: Data for tyske tanks

Day	OH Tanks	Damaged	Dst+Abnd	Total Loss
1	2500	0	0	0
2	2396	19	86	105
3	2367	69	48	117
4	2064	120	139	259
5	1754	100	215	315
6	1495	149	140	289
7	1406	77	80	157
8	1351	51	84	135
9	977	210	204	414
10	978	58	59	117
11	907	57	61	118
12	883	45	51	96
13	985	9	18	27
14	978	16	26	42
15	948	58	27	85
TOTAL		**1038**	**1238**	**2276**

Table 8. Soviet tank data. Dst+Abnd denotes destroyed and abandoned tanks. OH denotes the on hand amount.

Figure 8: Data for russiske tanks

8.2 Kode

8.2.1 Iminuit

```
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import scipy.optimize as sc
import scipy.stats as ss

from iminuit import Minuit
from iminuit.cost import LeastSquares

#Observerede vaerdier
G = [732, 611, 503, 364, 328, 265, 167, 110, 64, -15, -38, -45, -51]
R = [2278, 2019, 1704, 1415, 1258, 1123, 709, 592, 474, 378, 351, 309, 224]
t = np.arange(3, 16)

#Definering af koblede differentialligninger
def dydt(t, y, g, r):
    return [-r*y[1], -g*y[0]]

#Definering af goodness of fit parameter
```

```
def GoF(g, r):
    res = solve_ivp(dydt, [t[0], t[-1]], [G[0], R[0]], args=(g, r),
        t_eval=np.linspace(t[0], t[-1], len(t)))
    G0 = res.y[0]
    R0 = res.y[1]
    z1 = (G-G0)**2/G0
    z2 = (R-R0)**2/R0
    z  = np.append(z1,z2)
    return np.sum(z)

#Fit vha. Minuit-kommando
m = Minuit(GoF, g=573.48e-3, r=59.921e-3)
m.errordef = 1
q=m.migrad()
print(q)

#Forventede vrdier
t0 = res.t
G0 = res.y[0]
R0 = res.y[1]

#Plot af observerede- og forventede vaerdier
fig, ax = plt.subplots()
ax.plot(t0,G0,label='$\mathrm{G_E}$')
ax.plot(t0,R0,label='$\mathrm{R_E}$')
ax.plot(t,G,'ro',label='G')
ax.plot(t,R,'bo',label='R')
ax.set_xlabel('tid (dage)')
ax.set_ylabel('antal tropper')
ax.legend()
fig, ax = plt.subplots()
ax.set_xlabel('Antal russiske tropper (R)')
ax.set_ylabel('Antal tyske tropper (G)')
ax.plot(R,G,'ro',label='observerede antal')
ax.plot(R0,G0,'b-',label='forventede antal')
ax.legend()
```

8.2.2 Monte Carlo

$\pm\sigma$

```
import numpy as np
from scipy.integrate import solve_ivp
from scipy.linalg import norm
import matplotlib.pyplot as plt
from matplotlib import animation, rc
import scipy.optimize as sc

g=0.614
r=64.3e-3
ge=0.006
re=0.6e-3

import scipy.stats as ss
nrep = 100000

gmean=ss.norm.rvs(g,ge,nrep)
rmean=ss.norm.rvs(r,re,nrep)



gmin1=np.mean(gmean)-np.std(gmean)
gmax1=np.mean(gmean)+np.std(gmean)
rmin1=np.mean(rmean)-np.std(rmean)
rmax1=np.mean(rmean)+np.std(rmean)

k=[[g,r],[gmin1,rmin1],[gmax1,rmax1],[gmin1,rmax1],[gmax1,rmin1]]

G = [732, 611, 503, 364, 328, 265, 167, 110, 64, -15, -38, -45, -51]
R = [2278, 2019, 1704, 1415, 1258, 1123, 709, 592, 474, 378, 351, 309, 224]
t = np.arange(3, 16)

def dydt(t, y, g, r):
    return [-r*y[1], -g*y[0]]

fig, ax = plt.subplots(2,figsize=[7.5,11])

Rplot = ax[0].plot(t,R,'bo')
Gplot = ax[0].plot(t,G,'ro')
```

```
RG = ax[1].plot(R,G,'bo')


line1= [Rplot,Gplot]
line2= []
line3= [RG]
for i in [0,1,2,3,4]:
    res= solve_ivp(dydt, [t[0], t[-1]], [G[0], R[0]], args=k[i],
        t_eval=np.linspace(t[0], t[-1], 400))
    t0 = res.t
    G0 = res.y[0]
    R0 = res.y[1]
    plot1, = ax[0].plot(t0,G0)
    line1.append(plot1)
    plot2, = ax[0].plot(t0,R0)
    line1.append(plot2)
    plot3, = ax[1].plot(R0,G0)
    line3.append(plot3)




ax[0].legend(line1, ['R','G','1: $g = g$','2: $g = g-\sigma_g$','3: $g = g+
    \sigma_g$','4: $g = g-\sigma_g$','5: $g = g+\sigma_g$','1: $r = r$','2: $r =
    r-\sigma_r$','3: $r = r+ \sigma_r$','4: $r = r+\sigma_r$','5: $r = r-\sigma_r$'])
ax[1].legend(line3,['$g = g$, ' +'$r = r$','$g = g-\sigma_g$, ' +'$r =
    r-\sigma_r$','$g = g+\sigma_g$, ' +'$r = r+\sigma_r$','$g = g-\sigma_g$, ' +'$r =
    r+\sigma_r$','$g = g+\sigma_g$, ' +'$r = r-\sigma_r$'])
ax[0].set_xlabel('Tid(dage)')
ax[1].set_xlabel('Antal russiske tropper(R)')
ax[0].set_ylabel('Antal tropper')
ax[1].set_ylabel('Antal tyske tropper(G)')
```

8.2.3 χ^2 hypotesetest

```
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import scipy.optimize as sc
import scipy.stats as ss
```

```
from iminuit import Minuit
from iminuit.cost import LeastSquares

G = [732, 611, 503, 364, 328, 265, 167, 110, 64, -15, -38, -45, -51]
R = np.array([2278, 2019, 1704, 1415, 1258, 1123, 709, 592, 474, 378, 351, 309, 224])
t = np.arange(3, 16)

#Fit til data
def dydt(t, y, g, r):
    return [-r*y[1], -g*y[0]]

def GoF(g, r):
    res = solve_ivp(dydt, [t[0], t[-1]], [G[0], R[0]], args=(g, r),
        t_eval=np.linspace(t[0], t[-1], len(t)))
    G0 = res.y[0]
    R0 = res.y[1]
    z1 = (G-G0)**2/G0
    z2 = (R-R0)**2/R0
    z  = np.append(z1,z2)
    return np.sum(z)

m = Minuit(GoF, g=573.48e-3, r=59.921e-3)
m.errordef = 1
q=m.migrad()

res = solve_ivp(dydt, [t[0], t[-1]], [G[0], R[0]], args=m.values,
    t_eval=np.linspace(t[0], t[-1], 13))

t0 = res.t
G0 = res.y[0]
R0 = res.y[1]

#Liniarisering
c=4675/0.0643
x0=np.sqrt((R**2/c)-1)
xE=np.sqrt((R0**2/c)-1)
fig, ax = plt.subplots()
ax.plot(x0,G,'bo')
ax.plot(xE,G0,'r-')
ax.set_xlabel('x')
ax.set_ylabel('G')
```

```
#Chi2 hypotesetest
chmin=np.sum((xO[0:12]-xE[0:12])**2/xE[0:12])
p = ss.chi2.sf(chmin,12-1)
print(chmin)
print(p)
```