Om å kartleggja mørk materie med maskinlæring

Hans Georg Schaathun[1], Ben David Normann[2] og Kenny Solevåg-Hoti[3]

[1] georg@schaathun.net
[2] ben.d.normann@ntnu.no
Institutt for IKT og realfag
Fakultet for informasjonsteknologi og Elektroteknikk
[3] kenny.solevag-hoti@ntnu.no
Universitetsbiblioteket
NTNU — Noregs Teknisk-Naturvitskaplege Universitet
6025 Ålesund, Norway

Samandrag Gravitasjonslinsing er fenomenet der ljos frå fjerne himmellegeme vert avbøygd av tyngdekraften frå andre himmellegeme, som ofte ikkje er fullt synlege fordi mykje av massen er mørk materie. Observert gjennom ei gravitasjonslinse, framstår fjerne gallaksar som forvrengde. Der er mykje forskingsaktivitet som freistar å karleggja mørk materie ved å studera linseeffektar, men dei matematiske modellane er kompliserte og utrekningane krev i dag mykje manuelt arbeide som er svært tidkrevjande. I denne artikkelen drøftar me korleis me kan kombinera rouletteformalismen åt Chris Clarkson med maskinlæring for automatisk, lokal estimering av linsepotentialet i sterke linser, og me presenterer eit rammeverk med programvare i open kjeldekode for å generera datasett og validera resultat.

Keywords: gravitasjonslinsing · maskinlæring · rouletteformalisme · mørk materie · simulering

1 Innleiing

Eit av dei store måla i kosmologien er å kartleggja universet. Moderne teleskop gjev tilgang til eit enormt biletmateriale, men om lag 85% av massen er ikkje synleg. Sokalla mørk materie gjev ikkje frå seg ljos. Fysikarane veit lite om denne materien, men han må vera der for at den generelle relativitetsteorien skal stemma med observasjoner.

For å lokalisera den mørke materien, må ein sjå korleis han påverkar andre, synlege fenomen. Sidan ljoset vert påverka av tyngdekraften, vil mørk materie kunna danna sokalla gravitasjonslinser, som forvrenger bilete av meir fjerne galaksar. Dette er analogt til vanlege, optiske linser. Sjå t.d. Bertone og Tait (2018).

Ved å studera forvrengde bilete av fjerne gallaksar, er det råd å utleida plassering, form, og storleik på gravitasjonslinsene, men utrekningane er komplekse

og med dagens teknikkar tek det fleire dagar med manuelt arbeide å rekonstruera éi linse. Meir automatiserte teknikkar vil difor ha stor verdi. Hezaveh mfl. (t.d. 2017) demonstrerer linserekonstruksjon vha. maskinlæring, men dei tek berre for seg ein relativt enkel linsemodell og sterk linsing. Ofte bidreg mange galaksar til ein sokalla klyngelinse. Då får ein eit meir krevjande døme med fleire forvrengde bilete.

Konvensjonelle teknikker skil mellom svak og sterk linsing. Rouletteformalismen etter Clarkson (2016a) utvider svak linse-teknikkane for sterke linser. Dette gjev oss eit nytt og generelt rammeverk for gravitasjonslinser. Koeffisientane, eller amplitydane, i rouletteformalismen gjev ein lokal skildring av linsen og fortel korleis gravitasjonen verker rundt eitt punkt på himmelen. I denne artikkelen vil me skissera eit rammeverk for linserekonstruksjon vha. rouletteformalismen og maskinlæring, basert på roulettesimulatoren som me presenterte på ECMS 2023 (Schaathun mfl., 2023). Roulettemodellen og -simulatoren vert presentert i avsnitt 3. Me gjev eit konseptprov for å visa at maskinlæring er lovande (avsnitt 5), men i all hovudsak står dette att som eit ope problemet. Eit viktig bidrag er programvaren og infrastrukturen som skal gjera dette problemet tilgjengeleg for andre.

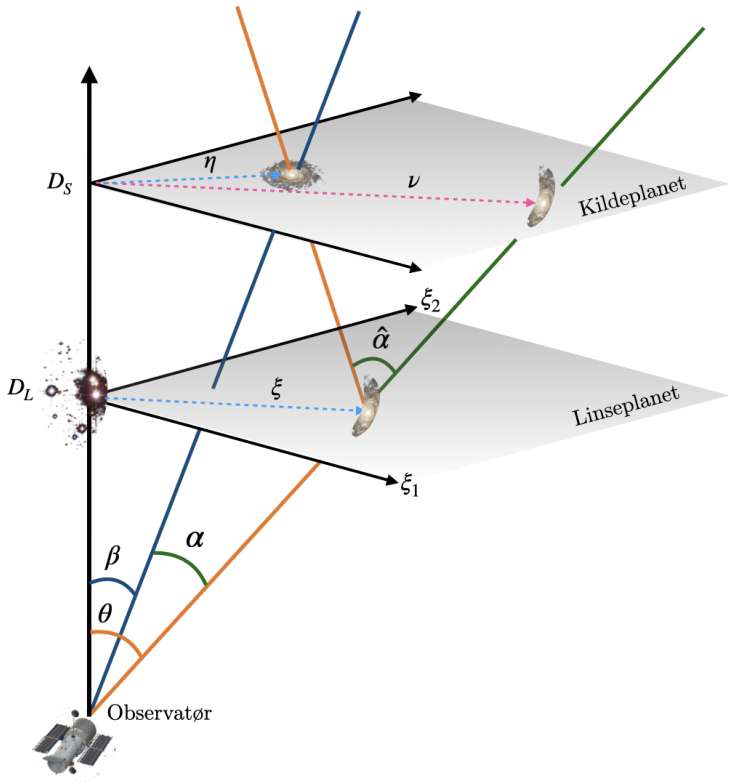
2 Modell og Problem

All materie, både ljos og mørk, oppfører seg som ei linse som forvrenger bilete av fjernare gallaksar. Eddington målte avbøyinga i solljoset under ei solformørking i 1919 og viste samsvar med den generelle relativitetsteorien etter Einstein. Omfattande teoretisk arbeide er gjort sidan den gongen, og til tross for periodar med pessimisme pga. skrinne observasjonar og dårleg oppløysing, er gravitasjonslinsing no vorte eit av dei mest lovande verktya for informasjonsmining frå nattehimmelen.

Figur 1 viser ein enkel modell med éin observasjon og éi linse. Massen er konsentrert i to plan, linseplanet som me søkjer å kartleggja og kjeldeplanet som er opphav til den synlege observasjonen. Me føreset at linsemassen er konsentrert i eit plan og ikkje har utstrekking langs synslina. Denne sokalla tynnlinsetilnærminga er rimeleg fordi avstandane er astronomiske og tjukkna dermed neglisjerbar. Strengt teke er linse- og kjeldeplanet kuleoverflater, men når synsvinkelen er smal kan me likevel føresetja flate plan (flathimmeltilnærminga). I det fylgjande lèt me D_L og D_S stå for kortaste avstand til hhv. linseplanet og kjeldeplanet.

Linsa dannar eit linsepotensial som er skildra som ein funksjon $\psi(\boldsymbol{\xi})$ som gjev ein reell verdi for kvart punkt $\boldsymbol{\xi}$ i linseplanet. I figur 1 ser me korleis observert ljos som kjem gjennom $\boldsymbol{\xi}$, ser ut til å koma frå $\boldsymbol{\nu}$ i kjeldeplanet, medan det i røynda kjem frå $\boldsymbol{\eta}$ og vert bøygd av linsa. Avbøyinga er bestemt av *raytrace*-likninga som er gjeven som

$$\boldsymbol{\eta} - \boldsymbol{\nu} = -D_\mathrm{S} D_\mathrm{L} \cdot \left(\frac{\partial \psi(\xi)}{\partial \xi_1}, \frac{\partial \psi(\xi)}{\partial \xi_2} \right). \tag{1}$$



Figur 1: Observasjon av nattehimmelen gjennom ei gravitasjonslinse.

Sidan avbøyinga er forårsaka av massen, må der vera ein samanheng mellom massetettleiken κ og linsepotensialet ψ. Denne samanhengen skriv me som

$$\kappa(\xi) = \frac{D_{\mathrm{L}}^2}{2}\nabla^2\psi(\xi) = \frac{D_{\mathrm{L}}^2}{2}\left(\frac{\partial^2\psi(\xi)}{\partial\xi_1^2} + \frac{\partial^2\psi(\xi)}{\partial\xi_2^2}\right). \tag{2}$$

Remark 1. Oppmerksame lesarar kan sjå at faktoren D_{L} er utelaten i programkoden, bortsett frå i forholdet $\chi = D_{\mathrm{L}}/D_{\mathrm{S}}$ som skalerer storleikar mellom linse- og kjeldeplanet. Dette er fordi D_{L} og D_{S} forøvrig vert kansellert i alle vidare utrekningar, noko som kan vera tidkrevjande men ganske rett fram å stadfesta.

Remark 2. Det er vanleg at linsa inneheld ljos materie i tillegg til mørk materie, slik at det er mogleg å fastsetja D_{L}, og me vil gå ut frå at dette er tilfellet og at både D_{L} og D_{S} er kjende størrelsar.

Example 1. Den singulære isotermiske sfæremodellen (SIS) (sjå t.d. Schneider mfl., 1992, Sec. 8.1.4) har linsepotensiale

$$\psi_{\mathrm{SIS}}(\boldsymbol{\xi}) = \frac{R_{\mathrm{E}}}{D_{\mathrm{L}}^2}|\boldsymbol{\xi}|, = \frac{R_{\mathrm{E}}}{D_{\mathrm{L}}^2}\sqrt{\xi_1^2 + \xi_2^2}, \tag{3}$$

der R_E er einsteinradien som gjev styrken (eller totalmassen) på linsa. *Raytrace*-likninga i SIS-modellen vert

$$\boldsymbol{\eta} = \left(1 - \frac{R_{\mathrm{E}}}{|\boldsymbol{\nu}|}\right)\boldsymbol{\nu}.$$

Sjølv om SIS er urealistisk, i og med at massen har uendeleg ustrekking, er modellen like fullt rekna som nyttig og mykje brukt i fysiske analysar.

Det er trivielt å simulera forvrengde bilete for ein gjeven kjelde- og linsemodell, i alle fall dersom ψ er deriverbar. For kvar piksel $\boldsymbol{\nu}$ i det forvrengde biletet, kan me rekna ut posisjonen $\boldsymbol{\eta}$ og *sampla* kjelda. Utfordringa er å finna ein plausibel linsemodell $\hat{\psi}$ for eit observert forvrengt bilete. Det generelle tilfellet, der ψ er ein vilkårleg funksjon, har uendeleg mange fridomsgradar og er dermed ikkje mogleg å løysa. Ei mogleg tilnærming er å føresetja ein konkret linsemodell (t.d. SIS) der ein berre treng å fastsetja nokon få parameter (som einsteinradien). Hezaveh mfl. (2017) brukte t.d. maskinlæring for å finna parametra i ein SIE-modell (singulær isotermisk ellipse). I praksis står me derimot gjerne overfor sokalla klyngelinser, der linsemassen er fordelt på fleire ulike objekt, kvar me sine parameter, og dette krev ein meir samansett modell.

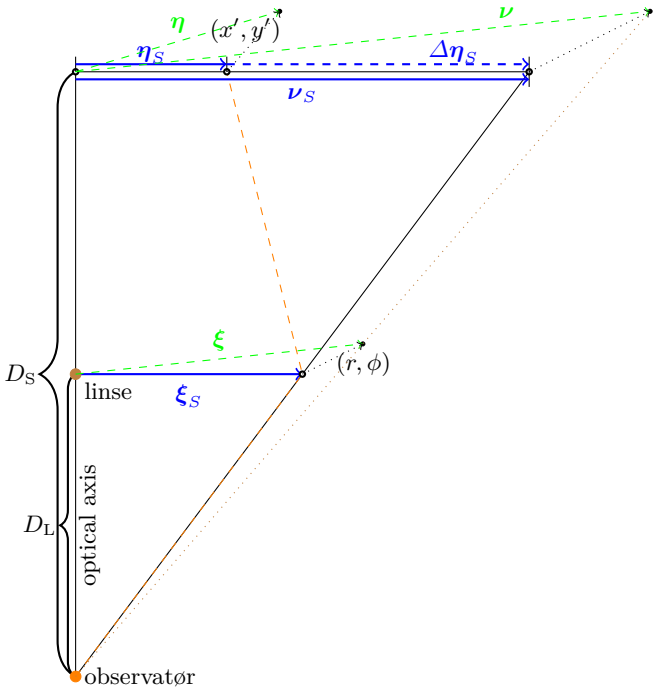
I dette arbeidet tek me ei alternativ tilnærming. I staden for å fastsetja linsepotensialet ψ over heile definisjonsområdet, freistar me å skildra ψ lokalt rundt eitt einskild punkt $\boldsymbol{\xi} = \boldsymbol{\nu} \cdot D_{\mathrm{L}}/D_{\mathrm{S}}$ i det forvrengte biletet. Dersom me kan gjera dette for fleire gallakser som er påverka av den same linsa, kan me i neste omgang freista å rekonstruera ψ globalt basert på fleire lokale skildringar.

3 Roulettemodellen

Rouletteformalismen vart introdusert av Clarkson (2016a) og utdjupa av Clarkson (2016b). I prinsippet er det ei taylorutviding av ψ rundt eit punkt $\boldsymbol{\xi}$ i linseplanet, og teknikken er godt etablert for svake linser der ein berre treng nokre få taylorledd for å få ei rimeleg nøyaktig tilnærming. Clarkson utvider denne teknikken for sterke linser ved å bruka fleire ledd, i kontrast til Fleury mfl. (2017) som utvider sterklinseteknikkar for svake linser.

For å simulera forvrengde bilete i rouletteformalismen, bruker me fyrst *raytrace*-likninga for å finna punktet $\boldsymbol{\xi}_S$ i linseplanet, som svarer til sentrum $\boldsymbol{\eta}_S$ i den opprinnelege kjelda. I SIS-modellen er $\boldsymbol{\xi}_S$ gjeven som

$$|\boldsymbol{\xi}_S| = \frac{D_{\mathrm{L}}}{D_{\mathrm{S}}} \cdot |\boldsymbol{\nu}_S| = \frac{D_{\mathrm{L}}}{D_{\mathrm{S}}} \cdot |\boldsymbol{\eta}_S| + R_{\mathrm{E}}.$$



Figur 2: Geometrien i den flate himmelen med dei kritiske punkta i roulettemodellen.

Punktet $\boldsymbol{\xi}_S$ bruker me som referansepunkt for rouletteutvidinga (jf. figur 2). Den fyrste rouletteamplityden (α_1^0, β_1^0) er då avbøyinga $\boldsymbol{\nu} - \boldsymbol{\eta}$,

$$\Delta\boldsymbol{\eta} = \boldsymbol{\nu} - \boldsymbol{\eta} = -D_\mathrm{S} \cdot (\alpha_1^0, \beta_1^0). \tag{4}$$

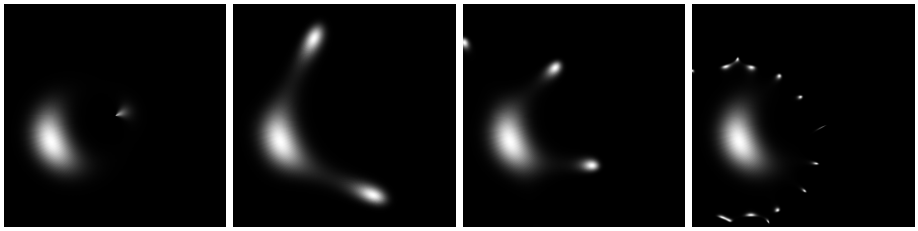
For å finna ljoset i eit vilkårleg punkt $\boldsymbol{\nu}$ nær $\boldsymbol{\nu}_S$, bruker me roulettelikninga. Me fylgjer Clarkson og skriv punktet $\boldsymbol{\xi}$ i polarkoordinatar (r, ϕ) med origo i $\boldsymbol{\xi}_S$, og $\boldsymbol{\eta}$ i kartesiske koordinatar (x', y') med origo i $\boldsymbol{\eta}_S$. Clarkson gjev oss då

$$\frac{D_\mathrm{L}}{D_\mathrm{S}} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix} = r \cdot \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix} + \sum_{m=1}^{\infty} \frac{r^m}{m! \cdot D_\mathrm{L}^{m-1}} \sum_{s=0}^{m+1} c_{m+s} \left(\alpha_s^m \boldsymbol{A}_s + \beta_s^m \boldsymbol{B}_s\right) \begin{bmatrix} C^+ \\ C^- \end{bmatrix} \tag{5}$$

der

$$C^\pm = \pm\frac{s}{m+1}, \tag{6}$$

$$c_{m+s} = \frac{1 - (-1)^{m+s}}{4} = \begin{cases} 0, & m+s \text{ er jamn}, \\ \frac{1}{2}, & m+s \text{ er odde}, \end{cases} \tag{7}$$



(a) *Raytrace* (b) Roulette $m \leq 3$ (c) Roulette $m \leq 5$ (d) Roulette $m \leq 15$

Figur 3: Simulering i roulettemodellen og samanlikning med *raytrace*.

og

$$\boldsymbol{A}_s = \begin{bmatrix} \cos(s-1)\phi & \cos(s+1)\phi \\ -\sin(s-1)\phi & \sin(s+1)\phi \end{bmatrix}, \tag{8}$$

$$\boldsymbol{B}_s = \begin{bmatrix} \sin(s-1)\phi & \sin(s+1)\phi \\ \cos(s-1)\phi & -\cos(s+1)\phi \end{bmatrix}. \tag{9}$$

Me kan leggja merke til at α_s^m og β_s^m forsvinn når $m+s$ er jamn. I svak linsing er det normalt tilstrekkeleg å bruka det nullte ($r(\cos\phi, \sin\phi)$) og fyrste ($m=1$) leddet i (5). Det nullte leddet gjev forstørringa medan det fyrste, kjend som skjæret (eller *shear*), gjev elongeringa. Den kjende Kaiser-Squires-likninga fortel oss korleis ein kan rekonstruera linsepotensialet ψ frå desse to ledda (Normann & Clarkson, 2020).

Ledda av høgare orden ($m > 1$) gjev informasjon om krumminga som oppstår i sterke linser. Enkelt sagt vil ein rund gallakse sjå oval ut bak ei svak linse. Bak ei sterk linse vert gallaksen krum som ein banan[4]. Svært sterke linseeffektar gjev ein sokalla einsteinring, der den fjerne gallaksen ser ut som ein ring som strekker seg rundt linsa. Når Clarkson utvider den kjende svaklinsemodellen for sterke linser, så reiser det spørsmålet om ogso Kaiser-Squires-likninga kan generaliserast.

Koeffisientane α_m^s og β_m^s i likningane over kallar me rouletteamplitydane. Normann og Clarkson (2020) har utleidd rekursive formlar for å finna algebraiske uttrykk for (α_m^s, β_m^s) som funksjonar av ξ for alle m og s. Implementasjonen vår, slik han vart presentert på ECMS 2023 (Schaathun mfl., 2023), gjorde to ting. For det fyrste reknar han ut rouletteamplitydane for SIS, og for det andre simulerer han forvrengde bilete ihht. likning (5).

Me kan samanlikna simulering i roulettemodellen med den eksakte *raytrace*-simuleringa i figur 3. Den uendelege rekkja i (5) må trunkerast i praksis, so me har simulert for $m \leq 3$, $m \leq 5$ og $m \leq 15$. Småbileta som ligg i ein ring rundt hovudbiletet er numeriske artifaktar. Ein kan visa at der alltid er $m+1$ slike bilete når m er odde og m når m er jamn, og dei legg seg som ein ring rundt $\boldsymbol{\nu}_S$. Radien i

[4] Eller kan henda meir som ei nyrebønne? Forfattarane er enno ikkje samde på dette punktet...

denne sokalla konvergensringen går mot $|\boldsymbol{\nu}_S|$ når m går mot uendeleg (Clarkson, 2016b). Det er sjølvsagt mogleg å maskera ut modellartifaktane sidan me veit kvar dei ligg.

Me kan merka oss at roulettemodellen ikkje får med bibiletet som ligg nær sentrum i *raytrace*-biletet. Bibiletet er eit resultat av ljos som går den lange vegen rundt linsa på motsett side. Dette bibiletet hamnar alltid utanfor konvergensringen i roulette[5] Bortsett fra bibiletet og dei falske bileta langs konvergensringen, ser med ei ei høveleg god tilnærming med fem ledd og med 15 ledd er biletet nær perfekt.

Remark 3. Der finst ei rekkje simulatorar for gravitasjonslinser. Særleg Lenstronomy (Birrer & Amara, 2018) og PyAutoLens (Nightingale mfl., 2021) er populære. Dei *tutorials* som finst for PyAutoLens gjev god innsikt i korleis ein kan byggja opp samansette linsemodellar og tilpasse parameter til empiriske bilete. Det som er nytt i tilnærminga vår er implementasjonen av rouletteformalismen og *lokal* skildring av linsepotensialet ogso for sterke linser.

4 Rekonstruksjonsmodell

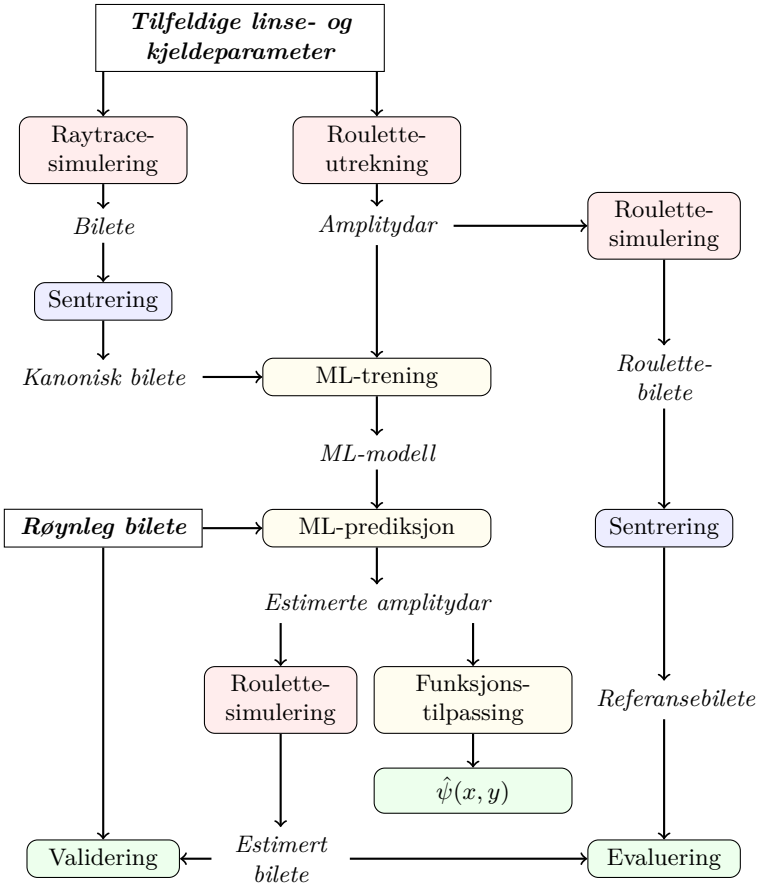
Føresetnaden for å kunna bruka ei generalisering av Kaiser-Squires, er at me kan estimera rouletteamplitydane frå observerte bilete. Det er mogleg for skjæret, men for høgare orden finst der ingen analytisk metode i dag. Simulatoren som me har drøfta over gjer det derimot mogleg å generera store datasett, og det er verd å sjå om maskinlæring kan estimera rouletteamplitydane. Figur 4 viser ein fullstendig prosess som kombinerer rouletteformalismen med maskinlæring.

Dei raude boksane bruker simulatoren *CosmoSim* som me har implementert, testa og gjort tilgjengeleg som open kjeldekode[6]. For å generera treningsdata til maskinlæringa bruker me *raytrace*-likninga (1) for å generera bilete (*input*) og roulette-utrekninga (sjå avsnitt 3) for å generera amplitydane som er *output* (*ground truth*) i maskinlæringa. Roulettesimulatoren gjer det mogleg å simulera forvrengde bilete direkte frå rouletteamplitydane, utan å ha nokon eksakt linsemodell. Dette gjev eit ekstra høve til å validera resultat, både under trening og i faktisk bruk. Bilete frå roulettesimuleringa gjev ei referansesanning som kan samanliknast med *input* til maskinlæringa. Dersom det estimerte biletet ikkje stemmar me det røynlege biletet, veit me at dei estimerte amplitydane er unøyaktige eller feile.

Me kjem tilbake til maskinlæringsdelen i avsnitt 5. Det som me har kalt funksjonstilpassing i figuren, svarer til Kaiser-Squires-likninga. Basert på estimerte rouletteamplitydar, ynskjer me å finna eit estimate $\hat{\psi}$ for linsepotensialet. Det er

[5] Det er mogleg bruka rouletteformalismen rundt eit anna punkt, og dermed teikna bibiletet, men sidan konvergensringen alltid går gjennom linsa, vil ein aldri kunna få med båe bileta i same roulettemodell.

[6] Publisert på github. Drøftinga her tek utgangspunkt i versjon 2.3.0: https://github.com/CosmoAI-AES/CosmoSim/releases/tag/v2.3.0.



Figur 4: Prosessmodell. Lokal rekonstruksjon av linsepotential.

viktig å merka seg at eitt sett med rouletteamplitydar berre gjev lokal informasjon om linsepotentialet rundt eitt punkt. Dersom me skal estimera ψ over heile definisjonsområdet vil me måtte bruka bilete av fleire forvrengde gallakser og rouletteamplitydane rundt kvar av dei. Dette er ofte mogleg i empiriske billete, der ein kan sjå mange gallaksar som er forvrengde av den same linsa. Denne funksjonstilpassinga står att som eit ope problem.

Bileta vert sentrerte for å unngå å lekkja informasjon om posisjonen til linsa relativt til den synlege gallaksen. Me reknar ljossentrum i biletet ved å ta gjennomsnitt av pikselindeksane vekta med ljosintensiteten. Biletet vert so translatert slik at ljossentrum vert sentrum i biletet. Dette gjev ein kanonisk form som me ogso kan finna for røynlege bilete

4.1 Rekonstruksjonssimulatoren

Som me har nemnd bruker *raytrace*-simulatoren eit koordinatsystem med origo på den optiske aksen (gjennom linsa). Når me simulerer med utgangspunkt i rouletteamplitydane er linsesentrum i prinsippet ukjend. Rett nok kan der vera noko ljos som gjev ein omtrentleg linseposisjon, men me vil ikkje gå ut frå at dette gjev ein tilstrekkeleg presis posisjon. Det er òg grunnen til at biletet vert sentrert rundt ljossentrum.

For å rekonstruera det forvrengde biletet treng me informasjon om kjelda, i tillegg til rouletteamplitydane som gjev informasjon om linsa. Kjeldeposisjonen er representert som $\boldsymbol{\xi}' = (\xi_1', \xi_2')$ relativt til origo i ljossentrum. For ei sfærisk kjelde treng me dessutan storleiken, som i *CosmoSim* er representert som standardavviket σ i ein gaussisk ljosfordeling. Andre kjeldemodellar krev fleire parameter, men det har me ikkje testa i denne studien.

Dei tre parametra σ, ξ_1' og ξ_2' vert inkludert i datasettet i maskinlæringa, saman med rouletteamplitydane.

4.2 Oversikt over implementasjonen

Sjølve simulatoren i *CosmoSim* er eit bibliotek i C++. To brukargrensesnitt er implementerte i Python. Kommandolinegrensesnittet som me bruker her, er designa for satsgenerering av bilete. GUI-verktyet drøfta me på ECMS (Schaathun mfl., 2023), og det er ikkje relevant her.

Simulatoren fylgjer ein enkel objektorientert struktur. Eit *Source*-objekt definerer den synlege gallaksen og genererer biletet slik det hadde sett ut utan gravitasjonslinsa. Her har me implementer underklasser for sfæriske og elliptiske linser, samt ein trefarga trekant til illustrasjonsformål.

Rouletteamplitydane vert rekna ut symbols, vha. sympy-biblioteket i python, og skrive til ei fil som vert lese i C++-biblioteket, som so evaluerer amplitydane i konkrete punkt.

Eit *Lens*-objekt definererer linsemodellen med alle dei analytiske eigenskapane som er kjende. Særleg er det ψ med dei to fyrsteordens partiellderiverte og rouletteamplitydane som trengst i simulatoren. Her har me førebels berre implementert ei underklasse for SIS, samt ei klasse som *samplar* ei anna linsemodell og

reknar med numerisk derivasjon. Sistnevnde kan vera nyttig til meir kompliserte linsemodellar der symbolsk derivasjon ikkje er mogleg innanfor rimeleg tid.

Sjølve simulatoren er ei underklasse av *LensModell*. Her har me implementert *RaytraceModel* og *RouletteModel*. Dei tek ei *Source* og ei *Lens* og bruker hhv. (1) og (5) for å transformera biletet frå *Source* til eit realistisk forvrengd bilete slik det vert observert. *RaytraceModel* hentar dei partiellderiverte av ψ frå *Lens*-objektet medan *RouletteModel* bruker rouletteamplitydane.

For å rekonstruera biletet frå rouletteamplitydane aleine, bruker me ei tredje *LensModel*-klasse, *RouletteRegenerator*, som ikkje bruker noko *Lens*-objekt. Simuleringa er den same som i *RouletteModel*, men rouletteamplitydane vert sette direkte i *RouletteRegenerator*, ingen annan informasjon om linsa vert tilgjengeleg.

Kommandolineprogrammet, slik me normalt bruker det, tek ein CSV-fil med linse- og kjeldeparameter og genererer eitt bilete per rad. I tillegg kan det, i same prosess, generera ei ny CSV-fil med rouletteamplitydane.

5 Resultat

For å demonstrera at rammeverket har noko for seg, presenterer me ein konkret serie av testar. Me har førebels gjort lite for å optimalisera maskinlæringsoppsettet eller utfallsrommoet for datasettet, og ein lyt difor lesa det som eit døme og ikkje som eit råd til endeleg løysing.

5.1 Biletgenerering

Datasettet simulerer ei sfærisk kjelde sett gjennom ei SIS-linse. Relativ avstand til linsa, $\chi = D_\mathrm{L}/D_\mathrm{S}$ set me konstant lik 0,5. Dette gjev fire variable parameter: storleiken (standardavviket) σ for kjelda, einsteinradien R_E, og kjeldeposisjonen som me skriv i polarkoordinatar (R, ϕ). Desse dreg me uniformt tilfeldig frå fylgjande sannsynsfordeling:

$$\begin{aligned} \sigma &\in \{1, 2, \dots 60\}, \\ R_E &\in \{5, 6, \dots 50\}, \\ \phi &\in \{0^\circ, 1^\circ, \dots 359^\circ\} \\ R &\in \{R_E, R_E + 1, \dots 100\} \end{aligned}$$

Polarkoordinatane vert rekna om til kartesiske koordinatar (x, y) som vert brukte i maskinlæringa, men me merker oss at det er avstanden R til origo som er uniformt fordelt, ikkje x- og y-koordinatane.

For å vurdera køyretid, har me køyrd sju samtidige satsar à 20 000 bilete. Kvar sats tek 59–65 minutt sanntid og 138–170 minutt CPU-tid på åtte kjernar[7]. Dette er overkommeleg, og langt fleire bilete enn me bruker i den vidare testen.

[7] Me har brukt tungreiningsklynga IDUN ved NTNU, og me har ikkje registrert kva node som har vore tildelt og dermed kjenner me ikkje prosessorspesifikasjonane.

Forsøk med fleire samtidige jobbar gjev lengre køyretid, og det er rimeleg å gå ut frå at flaskehalsen er skriving til disk.

Skriptet `datagen.py` genererer både bileta og CSV-filen med rouletteamplitydar; som fylgjar

```
python3 CosmoSimPy/datagen.py -D <biletkatalog> \
   -C -Z 800 -z 400 --lensmode SIS --modelmode Raytrace \
   --nterms 5 --outfile roulette.csv --csvfile dataset.csv \
   --xireference
```

Her lagar me bilete på 800×800 som vert klipt til 400×400 etter sentrering. Rouletteamplitydane vert funne for $m \leq 5$. Opsjonen `-xireference` seier at rouletteamplitydane vert rekna i den tilsynelatende posisjonen til sentrum i kjelda. Dette ligg normalt ikkje i ljossentrum av det forvrengde biletet.

Me simulerer dei forvrengde bileta i rouletteformalisem som referanse. Fordi denne simulatoren har eit anna API, vert dette gjort med eit eige skript, som fylgjer.

```
python3 CosmoSimPy/roulettegen.py -D <roulettebiletekatalog> \
        -n 5 -Z 400 --csvfile roulette.csv --xireference
```

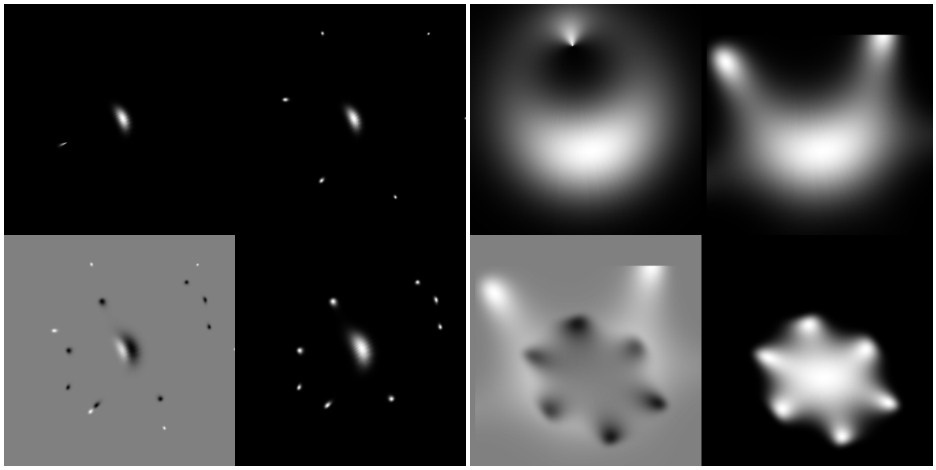
Køyretida er 29–30 minutt sanntid og 98–99 minutt CPU-tid per sats på 20 000 bilete.

Datasettet omfattar 33 søyler som trengst for å kunne resimulera det forvrengde biletet i rouletteformalismen. Der er 30 rouletteamplitydar for $m = 0, \ldots, 5$, samt linseparameteren σ og linseposisjonen (x', y') som vert registrert relativt sentrum i biletet (ljossentrum).

5.2 Maskinlæring

Til maskinlæringa har me brukt Inception v3, modifisert for å ta éin kanal (gråtone) inn og gje regresjonsdata ut, i staden for klassifisering. Modifikasjonane er tekne frå arbeidet åt Ingebrigtsen mfl. (2022). Optimeringsalgoritma er Adams der alle parameter har initialinstillingar bortsett frå læringsraten $\alpha = 0{,}0001$. Som tapsfunksjon bruker me gjennomsnittleg kvadratfeil (MSE). Me har brukt 4000 bilete til trening og 10 000 til testing. Med 50 epokar tek dette under to timar på ein NVIDIA A100 (GPU).

Figur 5 viser rekonstruerte bilete basert på estimerte amplitydar. I det eine biletet, der kjelda er stor, ser me ingen gjenkjennelege drag. I det andre, med ei mindre kjelde, ser me tydeleg at både retning frå origo og krumming er riktig, sjølv om rekonstruksjonen er langt frå nøyaktig. Me kan dermed slutta at det er mogleg å dra relevant informasjon ut av biletet vha. maskinlæring, og det er sannsynleg at ein kan finna betre resultat om ein legg arbeid i det.



(a) Døme på godt estimert bilete. (b) Døme på dårleg estimert bilete.

Figur 5: Samanlikning av rekonstruerte bilete. I kvar montage har me opprinneleg bilete øvst til høgre. Roulettesimulering frå faktiske amplitydar øvst til høgre og frå estimerte amplitydar nedzt til høgre. Nedst til venstre ser me differansen mellom dei to roulettesimuleringane.

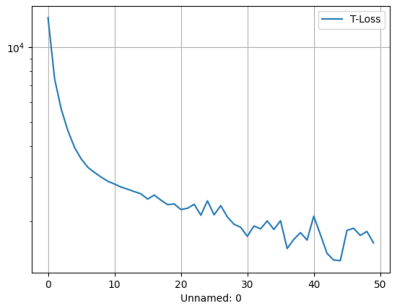
Ein kan stussa på at det rekonstruerte biletet ikkje har fleire falske bilete enn dei seks som me skulle ha sett for $m = 5$. Dette skuldast sannsynlegvis at dei estimerte amplitydane ikkje treng å svara til ein kontinuerleg og deriverbar funksjon ψ. Estimeringsfeilen kan gje artifaktar som ikkje er moglege i utgangspunktet.

Figur 6 viser kvantitativ evaluering av maskinlæringsoppsettet. Der er to ting som me skal merka oss. For det fyrste vert $\boldsymbol{\xi}'$ systematisk underestimert, noko som òg gjev forskyvinga i det rekonstruerte bilete i figur 5a. For det andre har med jamn forbetring i tapsfunksjonen i dei fyrste 12–14 epokane. Deretter tek tapsfunksjonen til å svinga. Det tyder på at læringsraten er for høg etter tolve epokar.

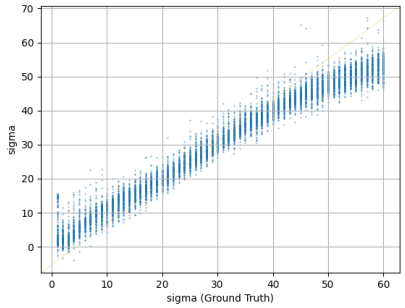
6 Vegen vidare

Me har etablert eit rammeverk for å arbeida med rekonstruksjon av rouletteamplitydar vha. maskinlæring, Dette opnar ei lang rekkje gode problem for vidare forsking. Innanfor maskinlæring gjenstår arbeidet med å finna god nettverksdesign og optimalisera hyperparameter.

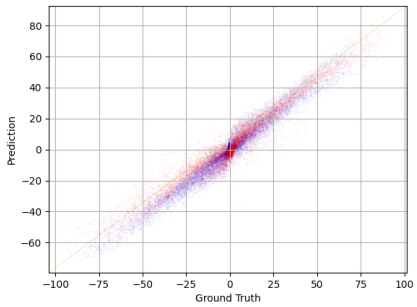
Me har framleis ikkje sett på testing med empiriske datasett. Dette er krevjande fordi astronomiske avstandar gjev låg oppløysing og ein må ta omsyn til optiske artifaktar i teleskopa. Tilretteleggjing av datasett frå røynda og tilpassing av treningssettet til empiriske data er den største og kanskje mest spanande utfordringa framover.



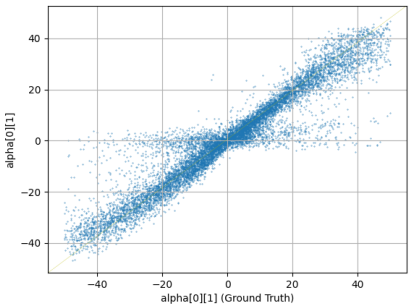
(a) Tapsfunksjonen under trening for kvar epoke.



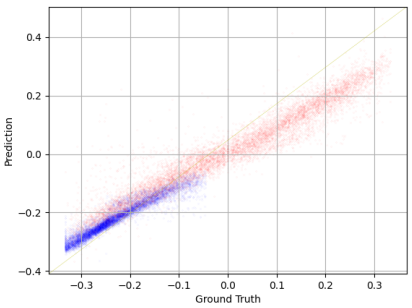
(b) Samanlikning av estimat og *ground truth* for σ.



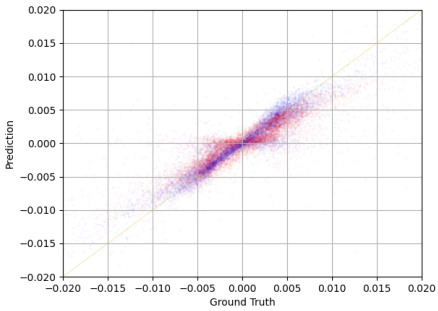
(c) Samanlikning av estimat og *ground truth* for ξ_1' (blått) og ξ_2' (raudt).



(d) Samanlikning av estimat og *ground truth* for α_1^0.



(e) Samanlikning av estimat og *ground truth* for α_0^1 (blått) og α_2^1 (raudt).



(f) Samanlikning av estimat og *ground truth* for α_1^2 (blått) og α_3^2 (raudt).

Figur 6: Evaluering av maskinlæringstesten.

Innanfor matematisk fysikk gjenstår arbeidet med å generalisera Kaiser-Squires eller utarbeida andre teknikkar for å rekonstruera linsepotentialet. Til sist vil det òg vera nyttig å utvida simulatoren med andre linse- og kjeldemodellar.

Referansar

Bertone, G. & Tait, T. M. P. (2018). A new era in the search for dark matter. *Nature*, *562*(7725), 51–56. https://doi.org/10.1038/s41586-018-0542-z

Birrer, S. & Amara, A. (2018). lenstronomy: Multi-purpose gravitational lens modelling software package. *Physics of the Dark Universe*, *22*, 189–201. https://doi.org/10.1016/j.dark.2018.11.002

Clarkson, C. (2016a). Roulettes: a weak lensing formalism for strong lensing: I. Overview. *Classical and Quantum Gravity*, *33*(16). https://doi.org/Artn16lt0110.1088/0264-9381/33/16/16lt01

Clarkson, C. (2016b). Roulettes: a weak lensing formalism for strong lensing: II. Derivation and analysis. *Classical and Quantum Gravity*, *33*(24), 245003. https://doi.org/10.1088/0264-9381/33/24/245003

Fleury, P., Larena, J. & Uzan, J. P. (2017). Weak Gravitational Lensing of Finite Beams. *Physical Review Letters*, *119*(19). https://doi.org/ARTN19110110.1103/PhysRevLett.119.191101

Hezaveh, Y. D., Levasseur, L. P. & Marshall, P. J. (2017). Fast automated analysis of strong gravitational lenses with convolutional neural networks. *Nature*, *548*(7669), 555–557. https://doi.org/10.1038/nature23463

Ingebrigtsen, S., Remøy, S. W., Runde, S. N. & Austnes, E. L. (2022). CosmoAI: A study of gravitational lensing through simulation and machine learning [Final year dissertation]. https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/3003634

Nightingale, J. W., Hayes, R. G., Kelly, A., Amvrosiadis, A., Etherington, A., He, Q., Li, N., Cao, X., Frawley, J., Cole, S., Enia, A., Frenk, C. S., Harvey, D. R., Li, R., Massey, R. J., Negrello, M. & Robertson, A. (2021). 'PyAutoLens': Open-Source Strong Gravitational Lensing. *Journal of Open Source Software*, *6*(58), 2825. https://doi.org/10.21105/joss.02825

Normann, B. D. & Clarkson, C. (2020). Recursion relations for gravitational lensing. *General Relativity and Gravitation*, *52*(3). https://doi.org/10.1007/s10714-020-02677-z

Schneider, P., Ehlers, J. & Falco, E. E. (1992). *Gravitational Lenses*. https://doi.org/10.1007/978-3-662-03758-4

Schaathun, H. G., Normann, B. D., Austnes, E. L., Ingebrigtsen, S., Remøy, S. W. & Runde, S. N. (2023). On the simulation of gravitational lensing [Florence, Italy, 21-23 June 2023]. I E. Vicario, R. Bandinelli, V. Fani & M. Mastroianni (Red.), *Proceedings of the 30th European Conference on Modelling and Simulation*. ECMS - European Council for Modelling; Simulation.