NOTES SUR LA DISPERSION SPECTRALE

Ariane LE CARDINAL, 30 mars 2021

1 Notations

Symbole	Description
ρ	fonction de pénalisation (Cauchy ou moindre carrés)
f	gaussienne
δ	carte des bad pixels (BP)
s^{cal}	facteur d'échelle
A	amplitude $\propto \frac{1}{2\pi\sigma^2}$
c_h^k	coefficients de la loi polynomiale (pixel)
c_h^λ	coefficients de la loi polynomiale (spectres)
x, y	position de la microlentille
xtrue, ytrue	paramètres vrais donnes a l'image simulée
xopt, yopt	paramètres optimisés
xinit, yinit	paramètres a l'initialisation
F(X)	valeur de la fonction de vraisemblance
‖G(X)‖	norme du gradient de la fonction de vraisemblance
$\lambda_{laser} = [\lambda_1, \lambda_2, \lambda_3, \lambda_4]$	tableau des longueurs d'onde de calibration
λ_0	moyenne de λ_{laser}

TABLE 1 – Notations

Le facteur d'échelle s^{cal} sert à ce que les interspectres ne soient pas détectés comme "outsiders" en diminuant la valeur d'ensemble des résidus. La fonction de pénalisation est ainsi robuste, car les pixels détectés comme défectueux sont très éloignés de la loi.

Le nombre d'acquisitions lasers n dépend du mode de calibration des bandes :

- YJ-mode : $n_\lambda^{cal} = 3$
- YH-mode : $n_\lambda^{cal} = 4$

Longueurs d'onde de calibration laser utilisées :

- $\lambda_1 = 987 : 72nm$
- $\lambda_2 = 1123 : 71nm$
- $\lambda_3 = 1309 : 37nm$
- $\lambda_4 = 1545 : 10nm$

2 Equations

On cherche à minimiser le terme de fidélité aux données $f(c_h^k, c_h^\lambda, \gamma_h, \Lambda)$ pour avoir une différence entre le modèle et les données la plus petite possible. Pour cela, on cherche les paramètres x, y, σ et A qui minimisent la fonction par ajustement.

Fonction de pénalisation : plusieurs choix sont possibles

- fonction de Cauchy : $\rho(r) = \frac{\gamma}{2} log(1 + \frac{r^2}{\gamma^2})$
  $\gamma = 2.385$

- fonction des moindres carrés :

Modèle utilisé :

Gaussienne :
$f(k_1, k_2, \sigma) = Aexp(\frac{(x_0-x)^2+(y_0-y)^2}{2\sigma^2})$

Calibration :
Data fidelity term :
$L(c_h^k, c_h^\lambda, \gamma_h, \Lambda) = \sum_{k=1}^{b} \rho\delta_k^{cal}(s^{cal-}1(d_p - \sum_{n=1} \xi_k(k_1, k_2, \sigma)))$

Position selon n $\in [1, 4]$:
x$(\lambda_n) = c_0 + c_1(\lambda_1 - \lambda_0) + c_2(\lambda_n - \lambda_0)^2$
$y(\lambda_n) = c_0' + c_1'(\lambda_1 - \lambda_0) + c_2'(\lambda_n - \lambda_0)^2$

Normalisation par λ_0 :
normalisation par λ_0 pour que l'optimsateur mette moins de temps. Mieux conditionne.
A l'origine : problème dans les unités (λ_0 en micron, donc coefficients en micron^{-1}).
Pas d'optimisation : c_0 sans unité, c_1 en 10^{-6}, c_2 en 10^{-12}.
$c_0 + c_1(\lambda_1 - \lambda_0) + c_2(\lambda_n - \lambda_0)^2$
Désormais : c_0 en pixel
$c_0 + c_1(\frac{\lambda_1-\lambda_0}{\lambda_0}) + c_2(\frac{\lambda_n-\lambda_0}{\lambda_0})^2$

3 Simulation d'une gaussienne

Packages utilisés :
TwoDimensional, Zygote, StatsBase, Plots, OptimPackNextGen, FITSIO

Fonctions et structures utilisées :

- Likelihood : renvoie $\sum$(données - modèle)^2
- Gradient : Dcrivée de likelihood, soit la dérivée de l'écart entre données et modèle.
  ∇cost = 0 au centre
  ∇cost > 0 pour fwhm + 2
- BoundingBox : délimite un cadre autour de la microlentille.

3.1 Simulation pour une microlentille

Méthode de simulation :
Comparaison d'une image générée avec les paramètres vrais (xtrue, ytrue,atrue, fwhmtrue) et une image finale data de paramètres optimises (xopt, yopt, aopt, fwhmopt). Si les images sont identiques, alors la simulation est correcte.

Input	Jeu de données
Simulation	a0 = 1. .- 0.2 .* randn(Float64,length(laser)); fwhm0= 5. .- 2. .* rand(Float64,length(laser)); C0[1 :2,1 :2] = [[6.2 10]; [25 80]]
Initialisation	ainit = a0 .+ (rand(Float64,laser.n) .- 0.5); fwhminit = fwhm0 .+ (rand(Float64,laser.n) .- 0.5); cinit[1 :2,1 :2] = [[6.2 0]; [25 60]]; xinit = vcat([ainit[:],fwhminit[:],cinit[:]]...)
Optimisation	aopt = xopt[1 :(laser.n)]; fwhmopt = xopt[(laser.n+1) :(2*laser.n)]; copt = reshape(xopt[(2*laser.n+1) :(4*laser.n)]; xopt = vmlmb(likelihood, xinit; verb=50, ftol=(0.0,0),gtol = (0.0,0))

TABLE 2 – Valeurs des coefficients en input

Résultats : On compare les trois paramètres initiaux a0, fwhm0 et Cinit aux paramètres optimisés aopt, fwhmopt.

Input	a	fwhm	C
Simulation	a0 = [1.0561309045643972, 1.152710401417176, 1.151099317026695]	fwhm0 = [3.043278514085292, 3.068569383978945, 3.4963421255308016]	C0 = [6.2 10.0 0.0; 25.0 80.0 0.0]
Initialisation	ainit = [0.6850303529100545, 0.6912655294445416, 0.9109697035079583]	fwhminit = [3.2273763932294592, 2.6525969218837857, 3.0415089372656503]	cinit = [6.2 0.0 0.0; 25.0 60.0 0.0]
Optimisation	aopt = [1.0297542706492195, 1.1341262000186536, 1.1894202502639288]	fwhmopt = [3.178884102908544, 3.076286963233277, 3.460936020233228]	copt = [6.167308613116124 9.801094424382724 -0.45938260019189386; 24.958919474842904 79.85987547044294 1.9442835603817556]

TABLE 3 – Résultats de la simulation

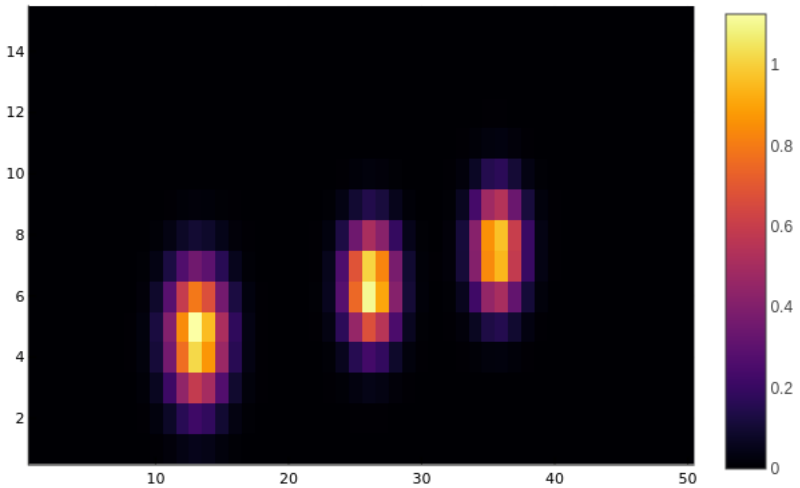

FIGURE 1 – Simulation d'une microlentille. Dans cette simulation d'une microlentille dans le mode YJ, on cherche a ajuster la loi de dispersion au trois spots gaussiens.

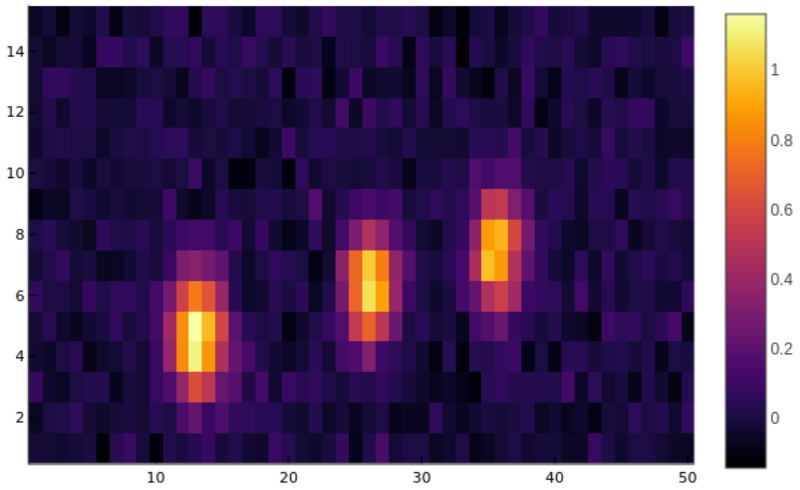

FIGURE 2 – Simulation de la microlentille avec ajout de bruit. On ajoute du bruit a la simulation pour chercher a retrouver les coefficients initiaux après optimisation.

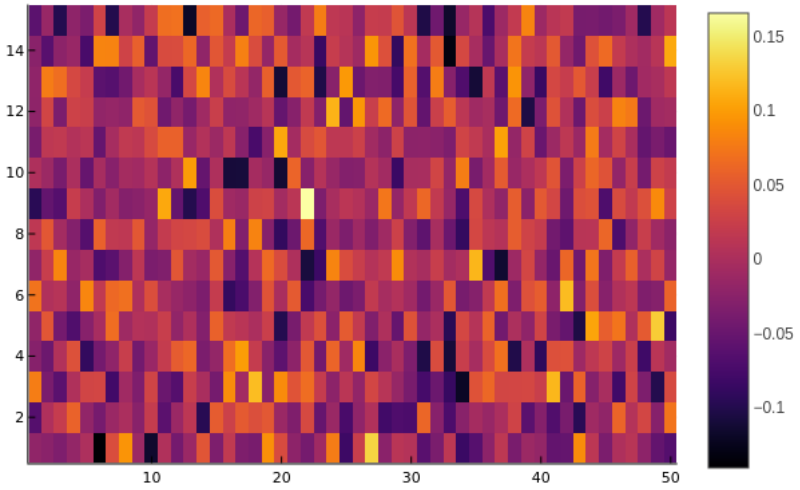

FIGURE 3 – Résidus

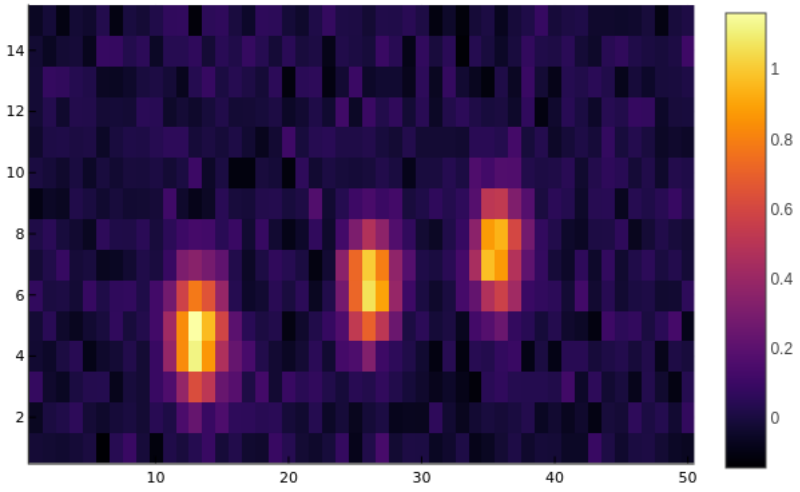

FIGURE 4 – Optimisation

3.2 Extraction d'une microlentille des fichiers de calibration HR4796-HD95086

Lire fichiers txt des coefficients c0 en x et en y : choisir une ligne dans les fichiers txt correspondant a une microlentille. Prendre la première ligne car pas de pixels défectueux.
Afficher la boundingbox associe au fichiers fits (IFS_ sim_ spec.fits).