CS 754 Homework 4

Waqar Mirza and Aditya Kudre

April 2022

1 Problem 1

1.1 Part a

Here is the plot of validation error $VE(g)$ v/s the logarithm of λ or $log(\lambda)$.

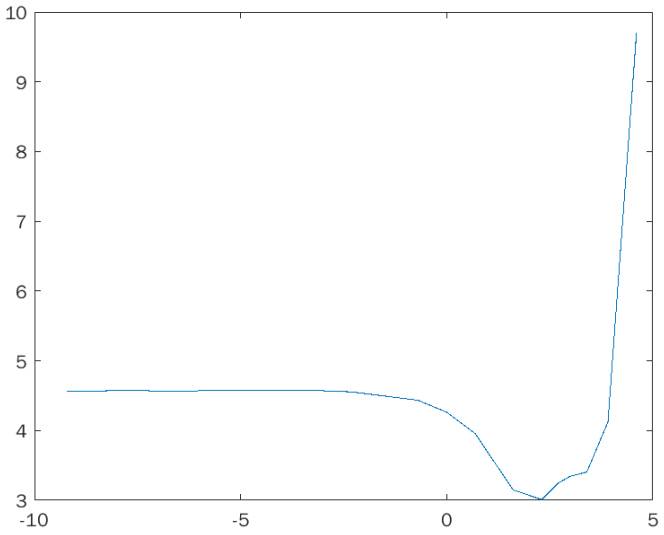

Figure 1: $VE(g)$ v/s $log(\lambda)$

As we can see from the plot, the lowest value of validation error occurs at $log(\lambda) = 2.3$ (approximately) or $\lambda = 10$. As seen from the graph, the error first decreases, reaches a minimum and then again increases. So, for values of $\lambda < 10$, our Algorithm reconstructs with the assumption of low sparsity which makes it work poorly. For high values of λ, a high sparsity is assumed which again causes the error to blow up.

Next, we have the plot of RMSE $||x_g - x||_2/||x||_2$ v/s $log(\lambda)$ as shown below.

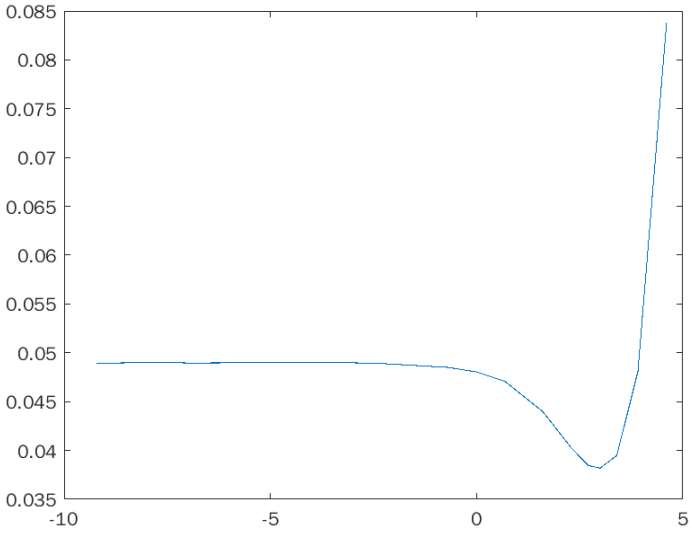

Figure 2: RMSE v/s $log(\lambda)$

From the graph, we can see that lowest RMSE occurs at $log(\lambda) = 3$ or $\lambda = 20$. This graph is also very similar to the validation error v/s $log(\lambda)$ graph. We can also see that the two values of λ for optimum performance match pretty well.

1.2 Part b

In cross validation, we are reconstructing using a randomly chosen set of measurements and then testing the reconstruction to check whether it matches well with rest of the measurements. This can be done perfectly (without any bias) only if the set used for reconstruction and that used for checking are disjoint sets. If they have some elements in common, we can expect an inherent bias while we are checking (validating) our reconstruction. If the validation and reconstruction sets overlap then cross validation will always give $\lambda = 0$.

1.3 Part c

The following theorem refers to the proxying ability of validation error.

Theorem 1 (Recovery error estimation) : Provided that m_{cv} is sufficiently large, with probability $erf(\frac{\lambda}{\sqrt{2}})$ the following holds.

$$h(\lambda, +)\epsilon_{cv} - \sigma_n^2 \leq \epsilon_x \leq h(\lambda, -)\epsilon_{cv} - \sigma_n^2 \tag{1}$$

where

$$h(\lambda, \pm) = \frac{m}{m_{cv}} \frac{1}{1 \pm \lambda\sqrt{\frac{2}{m_{cv}}}} \tag{2}$$

and

$$erf(u) = \frac{1}{\sqrt{\pi}} \int_{-u}^{u} e^{-t^2} dt \tag{3}$$

Here, ϵ_{cv} is the validation error while ϵ_x is the actual RMSE. m_{cv} is the number of measurements in the validation set and σ_n^2 is the standard deviation of the Gaussian Error. We can see that provided we have a very high λ or m_{cv}, we can expect ϵ_{cv} to be very close to ϵ_x.

1.4 Part d

The theorem in the book assumes a certain bound on the error term while if we do it with cross validation, there is no assumption on the error term. This makes cross validation a better approach to find λ. Basically, we need to have knowledge of the error in case of the theorem based approach while we do not need to know about the error term in case of cross validation. From the theorem we get

$$||x_g - x||_2 \leq \frac{3}{\gamma}\sqrt{k}\lambda_N \tag{4}$$

where $\lambda_N \geq 2||\Phi^T \eta||_\infty / N > 0$.

2 Problem 4

2.1 Details

- **Title :** Galaxy Image Classification using Non-Negative Matrix Factorization
- **Authors :** I.M.Selim, Arabi E. Keshk and Bassant M.El Shourbugy
- **Venue and Year of Publication :** International Journal of Computer Applications, March 2016
- **Link :** https://www.researchgate.net/publication/298803337_Galaxy_Image_Classification_using_Non-Negative_Matrix_Factorization

2.2 Further Details

The paper applies NMF to the classic problem of Galaxy Classification. Here, we are given an image of a galaxy and we are asked to classify it as either Spiral, Elliptical or Irregular (there are a few sub-types like barred spiral). First of all, we have the training set represented as V an $n \times m$ matrix which is factorized into an $n \times r$ matrix W and an $r \times m$ matrix H. Here, m is the number of examples in the training set and r is the number of classes in which we need to classify the galaxies. This is done by solving the following NMF problem.

$$min_{H,W} f(H,W) = \frac{1}{2}||V - WH||_F^2 \tag{5}$$

such that $H_{ij}, W_{ij} \geq 0$ for all i, j. In order to test it on unknown samples, we first make the test set S and then find a matrix A by solving the following NMF problem.

$$min_{A,W} f(A,W) = \frac{1}{2}||S - WA||_F^2 \tag{6}$$

Next, we look at the i^{th} column of A and find out the largest coefficient. The row index corresponding to this coefficient will give us the class of that unknown sample (by checking for the training sample that it corresponds to). Here, the idea is that the basis matrix W will contain information about the different features of different types of galaxies. This will eventually cluster the galaxies of same type together.

Problem 2:

(a) S_1 contains images obtained by applying a known derivative filter, say F_D, to images in S.

If we represent the images with vectors:

$S_1 = \{ F_D v \mid v \in S \}$

(here F_D is the matrix representing derivative filter (as it is a linear filter))

so now, we know

$v = D\theta \quad \forall \; v \in S$ where θ is sparse (say k sparse)

(θ depends on v)

So, $F_D v = F_D D \theta$

so $v' = F_D v = F_D D \theta$

where θ is k-sparse

so take $\boxed{\tilde{D} = F_D D}$ as the dictionary for class S_1, Then $\forall \; v' \in S_1$

$v' = \tilde{D}\theta$ for a k-sparse θ.

$\Rightarrow \tilde{D} = F_D D = F_D [D_1 \; D_2 \; \cdots \; D_m]$ (suppose m dictionary columns)

$= [F_D D_1 \quad F_D D_2 \; \cdots \; F_D D_m]$

So, the new dictionary columns are obtained by applying the derivative filter to old dictionary columns.

$(\tilde{D}_i = F_D D_i)$

(b) we can denote the rotation matrix for a vectorised image by angle α as R_α. (R_β for angle β)

now we know that $\forall\ v \in S,\ v = D\theta$, where θ is k-sparse

If v is rotated by angle α, we get $v' = R_\alpha v = R_\alpha D\theta$

and if v is rotated by β, $v' = R_\beta v = R_\beta D\theta$

so we can say that if we use the dictionary for S_2 as:

$\Rightarrow \tilde{D} = [R_\alpha D \mid R_\beta D]$

we will get a k-sparse representation for every $v' \in S_2$.

Since if v' is obtained by an α rotation, $v' = [R_\alpha D \mid R_\beta D] \begin{bmatrix} \theta \\ 0 \end{bmatrix} = \tilde{D}\tilde{\theta}$

here, $\tilde{\theta}$ is also k sparse. Similarly, if it is obtained by β rotation, $v' = [R_\alpha D \mid R_\beta D] \begin{bmatrix} 0 \\ \theta \end{bmatrix}$

again giving a k-sparse representation.

$\Rightarrow \boxed{\tilde{D} = [R_\alpha D \mid R_\beta D]}$

So final dictionary is union of rotated versions of D by α and β angles.

(c) In this part, we can write the intensity transformation for vectorised images with v from S and v' as the transformed image:

$$v' = \alpha (v \odot v) + \beta v + \gamma b$$

where $b_i = 1 \ \forall \ i \leq \dim(v)$ (i.e., b is a vector/image containing only ones)

$$b = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$ (here $\odot$ denotes pointwise multiplication)

now $v = D\theta$, θ is k-sparse.

so $v' = \alpha (D\theta \odot D\theta) + \beta D\theta + \gamma b$

$$= \alpha \left(\sum_i D_i \theta_i\right) \odot \left(\sum_j D_j \theta_j\right) + \beta D\theta + \gamma b$$

$$= \alpha \left(\sum_{i,j} (D_i \odot D_j)\, \theta_i \theta_j\right) + \beta D\theta + \gamma b$$

let D' be the dictionary containing $D_i \odot D_j \ \forall \ i, j \leq m$ (m is number of columns in D) as its columns.

so D' has $\frac{m(m+1)}{2}$ columns ($D_i \odot D_j = D_j \odot D_i$ is repeated if $i \neq j$) distinct

we can write

$$v' = \alpha D'\theta' + \beta D\theta + \gamma b$$

where θ' is a vectorised form of $\theta\theta^T$, i.e., $\theta' = \text{vec}(\theta\theta^T)$.

$$\Rightarrow v' = [b \mid D \mid D'] \begin{bmatrix} \gamma \\ \beta\theta \\ \alpha\theta' \end{bmatrix} = \tilde{D}\tilde{\theta}$$

note, $\alpha\theta'$ is k^2-sparse, $\beta\theta$ is k-sparse and γ is a scalar. So $\tilde{\theta}$ is $(k^2 + k + 1)$-sparse.

so use

$$\Rightarrow \boxed{\tilde{D} = [b \mid D \mid D']}$$ as the dictionary for S_3. where, $b = [1 \ 1 \ \dots \ 1]^T$

$D'_{(i+jm)} = D_i \odot D_j$ $\forall \ 1 \leq i, j \leq m$. Redundant columns are to be dropped.

(d) Suppose the blur kernel applied is b. [redacted] to image $v \in S$ to get v'!

$v = D\theta$, θ is k-sparse.

$$v' = b * (D\theta) = b * \left(\sum_i D_i \theta_i\right)$$

(* is the appropriate convolution)

$$= \sum_i (b * D_i)\, \theta_i$$

$$= [\, b * D_1 \;\; b * D_2 \;\; \ldots \;\; b * D_m \,]\, \theta$$

$$= \tilde{D}\, \theta$$

where $\tilde{D} = [\, b * D_1 \;\; b * D_2 \;\; \ldots \;\; b * D_m \,]$ should be chosen as the dictionary.

That is, $\tilde{D}$ is obtained by applying the same blur kernel b to the columns of dictionary D.

$$\Rightarrow \boxed{\tilde{D}_i = b * D_i}$$

(e) In this case the kernel b applied is some linear combination of known kernels $\{b_1, b_2, b_3, \ldots, b_n\} = B$.

So, $b = \sum_{i=1}^{n} \beta_i b_i$

So for a $v \in S$, we get $v' = b * v = b * \left(\sum_{i=1}^{m} D_i \theta_i\right)$

$$\Rightarrow v' = \left(\sum_{j=1}^{n} \beta_j b_j\right) * \left(\sum_{i=1}^{m} \theta_i D_i\right)$$

(convolution is linear)

$$= \sum_{i,j} \beta_j \theta_i (b_j * D_i)$$

$$= [\, b_1 * D \mid b_2 * D \mid \ldots \mid b_n * D \,]\, [\, \beta_1 \theta^T \mid \beta_2 \theta^T \mid \ldots \, \beta_n \theta^T \,]^T$$

where we define the notation:

$$b_i * D := [\, b_i * D_1 \;\; b_i * D_2 \;\; b_i * D_3 \ldots b_i * D_m \,]$$

So

$$v' = [b_1 * D \mid b_2 * D \mid b_3 * D \mid \ldots \mid b_n * D] \begin{bmatrix} \beta_1 \theta \\ \beta_2 \theta \\ \vdots \\ \beta_n \theta \end{bmatrix}$$

$$= \tilde{D}\,\tilde{\theta}$$

with $\tilde{D} = [b_1 * D \mid b_2 * D \mid \ldots \mid b_n * D]$

and $\tilde{\theta} = \begin{bmatrix} \beta_1 \theta \\ \vdots \\ \beta_n \theta \end{bmatrix}$. we see that $\tilde{\theta}$ is atmost $n\,k$ sparse.

So pick $\tilde{D}$ as the dictionary for S_5. That is, the new dictionary consists of union of all columns obtained by applying all blur kernels in B to the columns of D.

(f) We can represent the Radon transform matrix (with radon transform in angle θ) as R_θ.

So we obtain elements $\hat{v}'$ of S_6 from elements v of S as:

$$v' = R_\theta v$$

and $v = D\alpha$ where α is a k-sparse $m \times 1$ vector.

So

$$v' = R_\theta D \alpha \equiv \tilde{D}\alpha, \text{ where } \tilde{D} = R_\theta D$$

So $\tilde{D} = R_\theta D$ is chosen as dictionary of S_6, $\tilde{D}_i = R_\theta D_i$.

(In case there is reflection ambiguity we can also take reversed columns $(R_\theta D_i)_{rev}$ and include it in the dictionary)

So we get $\tilde{D}$ by taking radon transform of columns of D in the same direction θ.

(g) Let us denote the translation operation on a vectorised image v by offset $r = (x_0, y_0)$ as $T_r(v)$. Note that this translation operation is linear since $T_r(v_1 + v_2) = T_r(v_1) + T_r(v_2)$ and $T_r(\alpha v_1) = \alpha T_r(v_1)$

↳ This can be proved easily since on translation $I_{new}(x, y) = I_{old}(x - x_0, y - y_0)$

If $I_{old} = I_{1old} + I_{2old}$

so $I_{new}(x, y) = I_{1old}(x - x_0, y - y_0) + I_{2old}(x - x_0, y - y_0)$

$= I_{1new}(x, y) + I_{2new}(x, y)$

and if $I_{old} = \alpha I_{1old}$

$I_{new}(x, y) = \alpha I_{1old}(x - x_0, y - y_0)$

$= \alpha I_{1new}(x, y)$

So, we can denote this operation by a matrix multiplication $T_r v$, where T_r is the appropriate <u>translation matrix</u>.

now for $v \in S$, $v = D\theta$, θ is k-sparse

So if v is translated by $r_1 = (x_1, y_1)$ we get $v' = T_{r_1} v = T_{r_1} D \theta$

and if v is translated by $r_2 = (x_2, y_2)$ we get $v' = T_{r_2} v = T_{r_2} D \theta$

In both cases we have

$$v' = [T_{r_1} D \mid T_{r_2} D] \tilde{\theta}$$

where $\tilde{\theta} = \begin{bmatrix} \theta \\ 0 \end{bmatrix}$ for 1st case (with r_1) and $\tilde{\theta} = \begin{bmatrix} 0 \\ \theta \end{bmatrix}$ for case 2 (with r_2). So $\tilde{\theta}$ is also <u>k-sparse in both cases</u>. So take

$$\Rightarrow \boxed{\tilde{D} = [T_{r_1} D \mid T_{r_2} D]}$$ as the dictionary for S_7. Thus, the dictionary is formed by translating the members of dictionary D by (x_1, y_1) and (x_2, y_2) and taking the union of these translated members.

Page No.: YOUVA
Date:

Problem 3:

(1) $J(A_r) = \|A - A_r\|_F^2$, $A \in \mathbb{R}^{m \times n}$

A_r is rank r, $r < \min(m,n)$

consider the Singular Value Decomposition of A as:

$A = U D V^T$, $U \in \mathbb{R}^{m \times m}$, $V \in \mathbb{R}^{n \times n}$

U, V are unitary

and D contain the singular values $D_{11} \geq D_{22} \geq \dots \geq D_{\min(m,n),\min(m,n)}$ along its diagonal.

So now $J(A_r) = \|UDV^T - A_r\|_F^2$

$= \|U^T(UDV^T - A_r)V\|_F^2$

(since $\|X\|_F = \|UX\|_F$ for unitary U)

$\Rightarrow J(A_r) = \|D - (U^T A_r V)\|_F^2$

let take $D_r := U^T A_r V$

$\Rightarrow A_r = U D_r V^T$

So $J = \|D - D_r\|_F^2$

now as D is diagonal, we can minimise J by choosing the r largest singular values in D and placing them along the diagonal of D_r and keeping remaining elements of D_r as 0.

So

$D_r = \text{diag}(D_{11}, D_{22}, \dots, D_{rr}, 0, 0 \dots, 0)$

$D_r \in \mathbb{R}^{m \times n}$

So we get minimum value as:

$J(A_r) = \|D - D_r\|_F^2 = \boxed{\sum_{i>r} D_{ii}^2}$

as the minimum value.

and $\Rightarrow A_r = U D_r V^T = \boxed{\sum_{i=1}^{r} D_{ii} U_i V_i^T}$

This problem is well known as the low-rank approximation problem, and is used to obtain the best rank-r approximation to A. This is used in several places in image processing. One example is the KSVD algorithm, which uses this rank r approximation in the following step:
Let Y be the matrix of data points and A is the dictionary to be learnt. S is the sparse code matrix.
we try and $\min_{A,S} \|Y - AS\|_F^2$ with $\|s_i\|_0 \le T_0 \quad \forall i$

In an intermediate step, when we are updating the columns of A, we do:

$$\|Y - AS\|_F^2 = \|Y - \sum_{j=1}^{k} a_j s^j\|_F^2$$
$$= \|\left(Y - \sum_{j \neq k} a_j s^j\right) - a_k s^k\|_F^2 = \|E_k - a_k s^k\|_F^2$$

we approximate E_k with a rank-1 matrix $a_k s^k$ (it is rank 1 by construction).
so we need $a_k s^k = u_1 v_1^T \Lambda(1,1)$
when $E_k = U \Lambda V^T$ is the SVD of E_k.
So we take $a_k = u_1$, $s^k = \Lambda(1,1) v_1^T$.
In this way, this problem is used here.

(2) $J(R) = \|A - RB\|_F^2$, $A \in \mathbb{R}^{n \times m}$, $B \in \mathbb{R}^{n \times m}$ solution
$R \in \mathbb{R}^{n \times n}$,
and R is orthonormal. $\Rightarrow R^T R = R R^T = I$
we can write $\|X\|_F^2 = tr(X^T X)$

$$J(R) = tr((A^T - B^T R^T)(A - RB))$$
$$= tr(A^T A - B^T R^T A - A^T RB + B^T R^T RB)$$
$$= tr(A^T A) - tr(B^T R^T A) - tr(A^T RB) + tr(B^T B)$$
$$= tr(A^T A) + tr(B^T B) - 2\,tr(A^T RB)$$

[here we use $tr(XY) = tr(YX)$ and $tr(X) = tr(X^T)$]

$\Rightarrow J(R) = tr(A^TA) + tr(B^TB) - 2tr(BA^TR)$

To minimise $J(R)$ we need to **maximise** $tr(BA^TR)$.

let AB^T have the following SVD:

$AB^T = USV^T$, $U \in R^{n\times n}$, $V \in R^{n\times n}$

S contains the singular values in decreasing order on its diagonal. U and V are orthonormal.

So $tr(BA^TR) = tr(VS^TU^TR)$

$= tr(S^T(U^TRV))$

let $K = U^TRV$, K is also orthonormal $R^{n\times n}$ matrix

So we need to **maximise** $tr(S^TK) = \sum_i S_{ii} K_{ii}$

now since K is orthonormal $|K_{ii}| \leq 1$ and also $S_{ii} \geq 0$. So

$tr(S^TK) = \sum_i S_{ii} K_{ii} \leq \left(\sum_i S_{ii}\right) = tr(S)$

Equality occurs when $K = I$ is chosen. (all $K_{ii} = 1$)

So, we get the optimal R as:

$U^TRV = I \Rightarrow R = UV^T$

$\Rightarrow \boxed{R = UV^T}$ is the optimal R.

min value of $J(R)$ is then

$\min_R J(R) = \|A - UV^TB\|_F^2$.

This problem is well known as the orthogonal Procrustes problem and it appears in several places in image processing.

One example of its use is in Tomography under unknown angles in case of 3D images.

The solution to the problem is used in the following part of the tomographic reconstruction; The common lines between 3 projection planes needs to be estimated from the projections (without knowing projection angles), and planes need to be found. We write this mathematically for the i th plane as:

$$R_i \, C_i = B_i \longrightarrow 3\times 2 \text{ matrix of common lines in } i\text{th local co-ordinate system.}$$

R_i: corresponding rotation matrix.

C_i: 3×2 matrix of common lines for ith plane in global co-ordinates

Since there is noise and due to convenience we solve the following optimisation problem to obtain R_i.

$$\min_R E(R) \; ; \quad E(R) = \| RC_i - B_i \|_F^2 \, , \quad R^T R = I = RR^T$$

orthogonal

This is the Procrustes problem whose solution we just discussed. So if SVD of $B_i C_i^T = U D V^T$

$$\Rightarrow R = UV^T$$

Thus the problem makes its appearance here.

Date:

Problem 5:

We are given $y \sim Poi(I_0 \exp(-Rf))$

We can write

$$Pr(y|Rf) = \prod_i \left[\frac{[I_0 \exp(-(Rf)_i)]^{y_i}}{(y_i)!} \right] e^{-I_0 e^{-(Rf)_i}}$$

$$\left(\because Pr(x=k) = \frac{\lambda^k e^{-\lambda}}{k!} \text{ if } x \sim Poi(\lambda) \right)$$

$$\Rightarrow Pr(y|Rf) = \frac{I_0^{\sum_i y_i}}{\prod_i (y_i)!} \exp\left(-\sum_i y_i (Rf)_i\right) e^{-I_0 \sum_i e^{-(Rf)_i}}$$

we need to maximise $Pr(y|Rf)$, i.e., minimise $-\log(Pr(y|Rf))$.

$\Rightarrow$ same as minimising the following objective function:

$$J(f) = \sum_i y_i (Rf)_i + I_0 \sum \exp[(-Rf)_i]$$

$$= \boxed{y^T Rf + I_0 \sum_{i=1}^{m} \exp(-Rf)_i}$$

we need to solve for $\underset{f}{\text{argmin}}\, J(f)$ to reconstruct f given R, y.

If we have an additional i.i.d. additive gaussian noise with distribution $\mathcal{N}(0, \sigma^2)$.
Let y_p be the poisson R.V. and η be the gaussion noise.

$y_p \sim Poi(I_0 \exp(-Rf))$, $\eta_i \sim \mathcal{N}(0, \sigma^2)$

$y = y_p + \eta$

Date:

Now y_i is a continuous R.V. with distribution which is convolution of PDF of y_{p_i} and η_i.

~~$f_{y_i}(y_i | Rf) = \sum_{k \geq 0} Pr(y_{p_i} = k | Rf) f_\eta(y_i - k | k, Rf)$~~

$$f_{y_i}(y_i | Rf) = \sum_{k \geq 0} Pr(y_{p_i} = k | Rf) f_\eta(y_i - k | k, Rf)$$

$$= \sum_{k \geq 0} e^{-I_0 e^{-(Rf)_i}} \frac{I_0^k e^{-(Rf)_i k}}{k!} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(y_i - k)^2}{2\sigma^2}}$$

$$\Rightarrow f(y | Rf) \propto \prod_i \left[\sum_{k \geq 0} e^{-I_0 e^{-(Rf)_i}} \frac{I_0^k e^{-(Rf)_i k}}{k!} e^{-\frac{(y_i - k)^2}{2\sigma^2}} \right]$$

$$= \prod_i \left(\sum_{k \geq 0} \frac{I_0^k e^{-\left(I_0 e^{-(Rf)_i} + (Rf)_i k + \frac{(y_i - k)^2}{2\sigma^2}\right)}}{k!} \right)$$

$$= \prod_i g((Rf)_i, y_i)$$

where $\boxed{g(x, y) := \sum_{k \geq 0} \frac{I_0^k}{k!} \exp\left(-\left(I_0 e^{-x} + xk + \frac{(y-k)^2}{2\sigma^2}\right)\right)}$

we need to maximise $f(y | Rf)$, or minimise $-\log(f(y | Rf))$

so objective is:

$$\boxed{J(f) = \sum_i -\log(g((Rf)_i, y_i))}$$

and we need to carry out $\min_f J(f)$.