Completado de matrices de bajo rango mediante descenso de gradiente alternado

Eduardo Alexis Romo Almazán

I. Resumen

El problema de completado de matrices trata de recuperar una matriz a partir de un subconjunto de sus entradas, basándose en la interdependencia que existe entre las mismas, típicamente suponiendo que la matriz es de rango bajo.

A pesar de la naturaleza combinatoria del problema, existen muchos algoritmos eficientes computacionalmente para tratar el problema. Se presenta un algoritmo para completado de matrices de rango conocido.

II. Introducción

Las técnicas existentes para completado de matrices están fuertemente fundamentadas en la estructura de bajo rango de las matrices que se quieren completar, lo cual genera una interdependencia entre sus entradas. Buscar explícitamente la matriz de rango bajo consistente con las entradas conocidas se puede expresar como

$$\min_{Z\in\mathbb{R}^{m\times n}} \operatorname{rank} Z, \text{ sujeto a } P_\Omega(Z) = P_\Omega(Z_0), \tag{1}$$

donde $Z_0 \in \mathbb{R}^{m\times n}$ es la matriz que se quiere reconstruir, Ω es un subconjunto de índices de las entradas conocidas y P_Ω es el operador que muestrea únicamente las entradas en Ω.

El problema 1 es no convexo y generalmente NP-Completo. Uno de los enfoques más estudiados es reemplazar el rango por su versión relajada, la norma nuclear. De esta manera el problema se puede expresar como

$$\min_{Z\in\mathbb{R}^{m\times n}} \|Z\|_*, \text{ sujeto a } P_\Omega(Z) = P_\Omega(Z_0), \tag{2}$$

donde $\|Z\|_*$ representa la norma nuclear, dada por la suma de los valores propios de Z.

Muchos algoritmos para este problema involucran una descomposición SVD en cada iteración, lo cual limita su aplicabilidad para matrices muy grandes. Aquí se presenta una alternativa que evita el costo computacional de la SVD.

III. Desarrollo

III-A. Factorización de rango

Para evitar el costo computacional de calcular la SVD, supondremos que el rango r de la matriz que queremos completar es conocido y utilizaremos la descomposición mencionada en el siguiente teorema.

Teorema 1 (Factorización de rango)**.** *Dada una matriz $A \in \mathbb{R}^{m\times n}$ de rango r, existen matrices $C \in \mathbb{R}^{m\times r}$ y $F \in \mathbb{R}^{r\times n}$ tales que $A = CF$.*

Demostración. Como A es de rango r, existen r columnas de A linealmente independientes. Sea $c_1, c_2, \ldots, c_r$ una base del espacio generado por estas columnas y consideremos la matriz $C \in \mathbb{R}^{m\times r}$ cuyas columnas son c_i. Luego, cada columna de A es combinación lineal de las columnas en C. Si $a_1, a_2, \ldots, a_n$ son las columnas de A, entonces existen f_ij tales que

$$a_j = f_{1j}c_1 + f_{2j}c_2 + \cdots + f_{rj}c_r,$$

para cada j, por lo tanto $A = CF$, donde $F = [f_{ij}]$. □

Basada en esta simple factorización, en lugar de resolver el problema (1), resolveremos el problema

$$\min_{X,Y} \frac{1}{2}\|P_\Omega(Z_0) - P_\Omega(XY)\|_F^2. \tag{3}$$

Donde $\|\cdot\|_F$ es la norma de Frobenius, definida como

$$\|A\|_F^2 = \langle A, A\rangle_F = \sum_{i,j} A_{ij}^2$$

$$\langle A, B\rangle_F = \sum_{i,j} A_{ij}B_{ij} = \operatorname{tr}(A^T B)$$

III-B. Minimización alternada

Consideremos una función $f(X,Y)\colon \mathcal{X}\times\mathcal{Y} \to \mathbb{R}$. El método de minimización alternada minimiza $f(X,Y)$ sucesivamente sobre X y Y.

El algoritmo PowerFactorization(PF), es un algoritmo que resuelve el problema (3) utilizando esta idea.

Input: $P_\Omega(Z^0),\ X_0 \in \mathbb{R}^{m\times r}, Y_0 \in \mathbb{R}^{r\times n}$
Repeat
1. Fix Y_i , solve $X_{i+1} = \arg\min_X \|P_\Omega(Z^0) - P_\Omega(XY_i)\|_F^2$
2. Fix X_{i+1}, solve $Y_{i+1} = \arg\min_Y \|P_\Omega(Z^0) - P_\Omega(X_{i+1}Y)\|_F^2$
3. $i = i + 1$

Until termination criteria is reached
Ouput X_i, Y_i

Figura 1. PowerFactorization

Notemos que en cada subproblema hay que resolver un problema de mínimos cuadrados, lo cual determina la complexidad del algoritmo. Existen múltiples algoritmos que se basan en este y tratan de reducir el costo computacional de mínimos cuadrados, utilizando distintas alternativas.

La alternativa que vamos a seguir para reducir el costo computacional de PowerFactorization es reemplazar resolver el problema de mínimos cuadrados por un único paso en la dirección de máximo descenso.

III-C. Producto de Hadamard

Para calcular los gradientes utilizaremos el producto de Hadamard, definido para dos matrices $A, B \in \mathbb{R}^{m\times n}$ como

$$A \circ B = [a_{ij}b_{ij}]_{ij}$$

Observemos que este producto es conmutativo, asociativo y distributivo sobre la suma. Además cumple las siguientes propiedades:

$$(M \circ A)^T = [m_{ij}a_{ij}]_{ij}^T = [m_{ji}a_{ji}]_{ij} = M^T \circ A^T$$

Si M es una matriz binaria, es decir, sus entradas son 0 ó 1:

$$\begin{aligned}\operatorname{tr}\left[(M \circ A)^T(M \circ B)\right] &= \sum_{i,j} m_{ij}^2 a_{ij}b_{ij} \\ &= \sum_{i,j} m_{ij}a_{ij}b_{ij} \\ &= \operatorname{tr}\left[(M \circ A)^T B\right] \\ &= \operatorname{tr}\left[A^T(M \circ B)\right]\end{aligned}$$

Sea $M = [\mathbb{1}_\Omega(i,j)]_{ij}$. Denotando $W = XY$, podemos reescribir $f(X,Y)$ como

$$\begin{aligned}f(X,Y) &= \frac{1}{2}\operatorname{tr}\left[P_\Omega(Z_0 - W)^T P_\Omega(Z_0 - W)\right] \\ &= \frac{1}{2}\operatorname{tr}\left[(M \circ (Z_0 - W))^T(M \circ (Z_0 - W))\right] \\ &= \frac{1}{2}\operatorname{tr}\left[(M^T \circ (Z_0^T - W^T))(Z_0 - W)\right] \\ &= \frac{1}{2}\operatorname{tr}\left[(M \circ Z_0)^T Z_0\right] - \operatorname{tr}\left[(M \circ Z_0)^T XY\right] \\ &\quad + \frac{1}{2}\operatorname{tr}\left[(M \circ XY)^T XY\right] \qquad (4)\end{aligned}$$

III-D. Gradientes y tamaños de paso exactos

Para calcular los gradientes, observemos que

$$\frac{\partial}{\partial x_{rs}}\operatorname{tr}(AX) = \frac{\partial}{\partial x_{rs}}\sum_{i,k} a_{ik}x_{ki} = a_{sr},$$

por cual $\nabla_X\left(\operatorname{tr}(AX)\right) = A^T$. Utilizando esto, obtenemos los gradientes del segundo término de la ecuación (4).

$$\begin{aligned}\nabla_X\left(\operatorname{tr}\left[(M \circ Z_0)^T XY\right]\right) &= (M \circ Z_0)Y^T \\ \nabla_Y\left(\operatorname{tr}\left[(M \circ Z_0)^T XY\right]\right) &= X^T(M \circ Z_0)\end{aligned}$$

Para derivar el tercer término denotamos

$$h(X,Y) = \sum_{i,j} m_{ij}w_{ij}^2, \qquad w_{ij} = \sum_k x_{ik}y_{kj},$$

y calculamos las derivadas parciales

$$\begin{aligned}\frac{1}{2}\frac{\partial h}{\partial x_{rs}} &= \sum_j m_{rj}\left(\sum_k x_{rk}y_{kj}\right)y_{sj} \\ &= \sum_j (M \circ XY)_{rj}\left(Y^T\right)_{js} \\ \frac{1}{2}\frac{\partial h}{\partial y_{rs}} &= \sum_i m_{is}\left(\sum_k x_{ik}y_{ks}\right)x_{ir} \\ &= \sum_i \left(X^T\right)_{ri}(M \circ XY)_{is}\end{aligned}$$

Con lo que obtenemos

$$\begin{aligned}\frac{1}{2}\nabla h(X) &= (M \circ XY)\,Y^T \\ \frac{1}{2}\nabla h(Y) &= X^T(M \circ XY)\end{aligned}$$

De la ecuación (4) concluimos que los gradientes de f son

$$\begin{aligned}\nabla f(X) &= -(M \circ Z_0)Y^T + (M \circ XY)Y^T \\ &= -[M \circ (Z_0 - XY)]Y^T \\ &= -\left[P_\Omega(Z_0 - XY)\right]Y^T \\ \nabla f(Y) &= -X^T(M \circ Z_0) + X^T(M \circ XY) \\ &= -X^T[M \circ (Z_0 - XY)] \\ &= -X^T\left[P_\Omega(Z_0 - XY)\right]\end{aligned}$$

Se pueden encontrar los tamaños de paso de manera exacta. Para esto definamos

$$g_X(t) = \frac{1}{2}\|P_\Omega(Z_0) - P_\Omega((X - t\nabla f(X))Y)\|_F^2$$

y notemos que $t_x = \operatorname{argmin} g_X(t)$. Para derivar reescribimos $g_X(t)$ como sigue. Denotamos $Q = Z_0 - XY$ y $G = \nabla f(X)Y$.

$$\begin{aligned}g_X(t) &= \frac{1}{2}\|M \circ (Q + tG)\|_F^2 \\ &= \frac{1}{2}\operatorname{tr}\left[(M \circ (Q + tG))^T(M \circ (Q + tG))\right] \\ &= \frac{1}{2}\operatorname{tr}\left[(M \circ Q)^T Q + 2t(M \circ Q)^T G + t^2(M \circ G)^T G\right]\end{aligned}$$

Con lo que obtenemos

$$\begin{aligned}g_x'(t) &= \operatorname{tr}\left[(M \circ Q)^T G + t(M \circ G)^T G\right] \\ &= \operatorname{tr}\left[(M \circ (Q + tG))^T G\right] \\ &= \langle P_\Omega(Z_0) - P_\Omega((X - t\nabla f(X))Y), \nabla f(X)Y\rangle_F \\ &= \langle P_\Omega(Z_0) - P_\Omega(XY), \nabla f(X)Y\rangle_F \\ &\quad + t\langle P_\Omega(\nabla f(X)Y), \nabla f(X)Y\rangle_F \\ &= \langle (P_\Omega(Z_0) - P_\Omega(XY))Y^T, \nabla f(X)\rangle_F \\ &\quad + t\langle P_\Omega(\nabla f(X)Y), P_\Omega(\nabla f(X)Y)\rangle_F \\ &= -\langle \nabla f(X), \nabla f(X)\rangle_F + t\langle P_\Omega(\nabla f(X)Y), P_\Omega(\nabla f(X)Y)\rangle_F\end{aligned}$$

Igualando lo anterior a cero obtenemos

$$t_x = \frac{\|\nabla f(X)\|_F^2}{\|P_\Omega(\nabla f(X)Y)\|_F^2}$$

De manera análoga se obtiene

$$t_y = \frac{\|\nabla f(Y)\|_F^2}{\|P_\Omega(X\nabla f(Y))\|_F^2}$$

El algoritmo queda como se muestra en la figura 2.

Input: $P_\Omega(Z^0),\ X_0 \in \mathbb{R}^{m\times r},\ Y_0 \in \mathbb{R}^{r\times n}$
Repeat
1. $\nabla f_{Y_i}(X_i) = -(P_\Omega(Z^0) - P_\Omega(X_iY_i))Y_i^T,\ t_{x_i} = \frac{\|\nabla f_{Y_i}(X_i)\|_F^2}{\|P_\Omega(\nabla f_{Y_i}(X_i)Y_i)\|_F^2}$
2. $X_{i+1} = X_i - t_{x_i}\nabla f_{Y_i}(X_i)$
3. $\nabla f_{X_{i+1}}(Y_i) = -X_{i+1}^T(P_\Omega(Z^0) - P_\Omega(X_{i+1}Y_i)),\ t_{y_i} = \frac{\|\nabla f_{X_{i+1}}(Y_i)\|_F^2}{\|P_\Omega(X_{i+1}\nabla f_{X_{i+1}}(Y_i))\|_F^2}$
4. $Y_{i+1} = Y_i - t_{y_i}\nabla f_{X_{i+1}}(Y_i)$
5. $i = i + 1$

Until termination criteria is reached
Output: X_i, Y_i

Figura 2. Alternating Steepest Descent

III-E. Actualización eficiente de residuos

Cuando se actualiza X_i a X_{i+1} hay que recalcular el residuo $P_\Omega(Z_0) - P_\Omega(X_{i+1}Y)$ y respectivamente lo mismo cuando se actualiza a Y_{i+1}. Se puede evitar el costo del producto de matrices con las siguientes propiedades:

$$\begin{aligned} P_\Omega(X_{i+1}Y_i) &= P_\Omega\left((X_i - t_{x_i}\nabla f_{Y_i}(X_i))\,Y_i\right) \\ &= P_\Omega(X_iY_i) + t_{x_i}P_\Omega\left(\nabla f_{Y_i}(X_i)Y_i\right) \\ P_\Omega(X_{i+1}Y_{i+1}) &= P_\Omega\left(X_{i+1}(Y_i - t_{y_i}\nabla f_{X_{i+1}}(Y_i))\right) \\ &= P_\Omega(X_{i+1}Y_i) + t_{y_i}P_\Omega\left(X_{i+1}\nabla f_{X_{i+1}}(Y_i)\right) \end{aligned}$$

Notemos que las cantidades $P_\Omega\left(\nabla f_{Y_i}(X_i)Y_i\right)$ y $P_\Omega\left(X_{i+1}\nabla f_{X_{i+1}}(Y_i)\right)$ ya se calcularon antes en los denominadores de t_{x_i} y t_{y_i}, por lo que ya no hay que hacer ningún producto matricial.

IV. Experimentos

Para probar el algoritmo vamos a usar la aproximación de una imagen en blanco y negro con otra imagen, que vista como matriz tenga un rango más bajo. Para esto usamos el siguiente teorema.

Teorema 2 (Eckart-Young-Mirsky)**.** *Sea $A \in R^{m\times n}$ con $m \geq n$ de rango r y sea $A = U\Sigma V^T$ la descomposición en valores singulares de A. La matriz de rango k que mejor aproxima a A en la norma de Frobenius está dada por*

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T,$$

donde u_i y v_i denotan la $i-$ésima columna de U y V, respectivamente.

Demostración. Sea $B \in \mathbb{R}^{m\times n}$ una matriz de rango k. Observemos que como U y V son matrices ortogonales

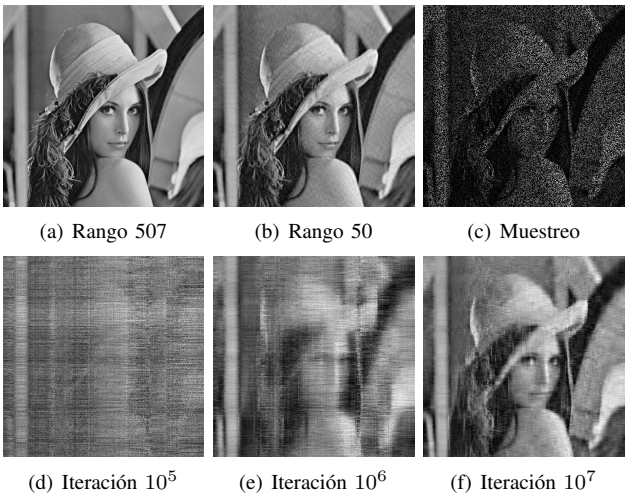
$$\begin{aligned} \|A - B\|_F^2 &= \|U\Sigma V^T - B\|_F^2 \\ &= \|\Sigma - U^TBV\|_F^2 \\ &= \|\Sigma - N\|_F^2 \end{aligned}$$

donde $N = U^TBV$ es de rango k. Expandiendo lo anterior tenemos

$$\begin{aligned} \|\Sigma - N\|_F^2 &= \sum_{ij}(\Sigma_{ij} - N_{ij})^2 \\ &= \sum_{i\leq r}(\sigma_i - N_{ii})^2 + \sum_{i>r}N_{ii}^2 + \sum_{i\neq j}N_{ij}^2 \end{aligned}$$

La expresión anterior se minimiza cuando las entradas de N fuera de la diagonal son cero y N_{ii} son cero para $i > r$. Por último, queda minimizar la primera suma sujeto a que exactamente k de los N_{ii} sean distintos de cero. Esto ocurre cuando $N_{ii} = \sigma_i$ para $i \leq k$ y los demás son cero. Finalmente, de la relación $B = UNV^T$ se obtiene el resultado. □

Utilizamos este algoritmo con 2 imágenes de dimensiones 512×512, que aproximamos por otras de rango 50. Las imágenes de rango 50 son muestreadas, tomando un $35\,\%$ de sus entradas de manera uniforme. El resultado de este muestreo es el input del algoritmo, para inicializar las matrices X y Y se utilizan matrices con valores aleatorios enteros entre 0 y 255. En las figuras 3 y 4 se muestran los resultados obtenidos. En la figura 5 se comparan las iteraciones contra el residuo relativo $\|P_\Omega(Z_0 - XY)\|/\|P_\Omega(Z_0)\|$. Para no utilizar demasiada memoria, estos residuos se calcularon únicamente cada 1000 iteraciones.



(a) Rango 507 (b) Rango 50 (c) Muestreo

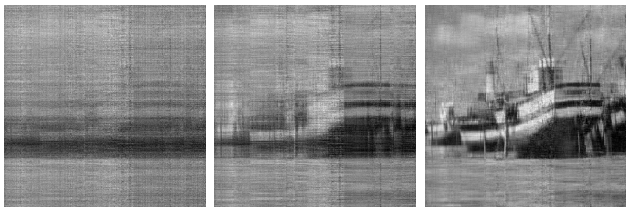
(d) Iteración 10^5 (e) Iteración 10^6 (f) Iteración 10^7

Figura 3. Lena

V. Conclusiones

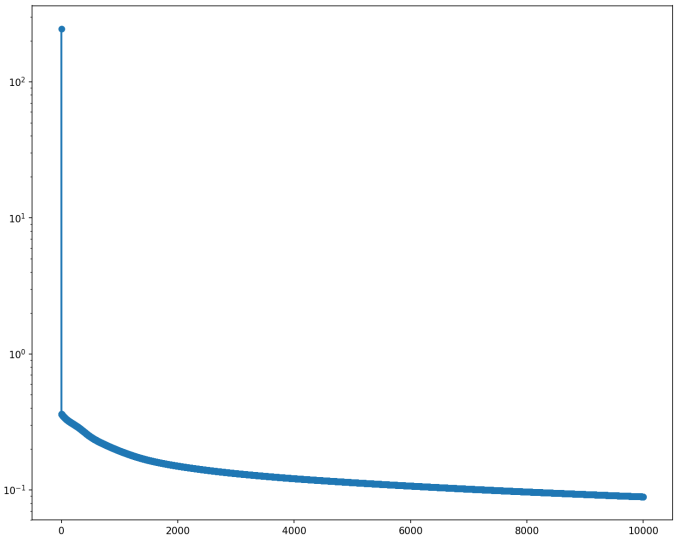
A pesar de la simplicidad del algoritmo, obtuvimos buenos resultados en términos de los residuos obtenidos. La convergencia del algoritmo es lenta, sin embargo los residuos siempre son decrecientes. Se pueden buscar formas de implementar el algoritmo para mejorar el tiempo, como una implementación en C++ con matrices ralas.

(a) Rango 512 (b) Rango 50 (c) Muestreo

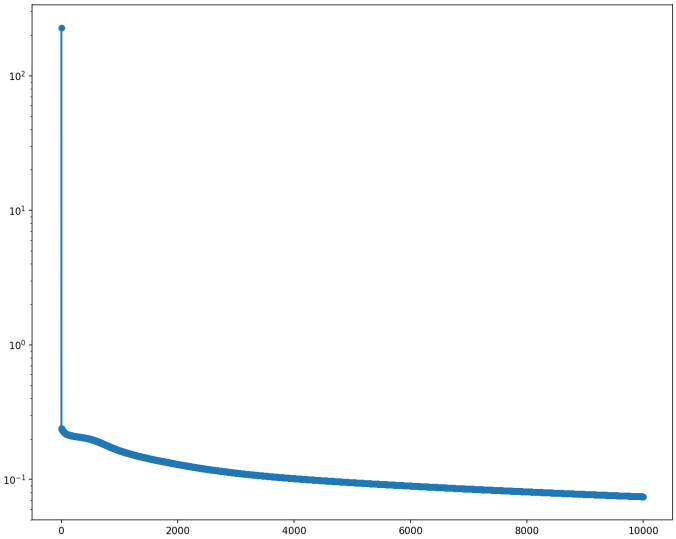


(d) Iteración 10^5 (e) Iteración 10^6 (f) Iteración 10^7

Figura 4. Boat



(a) Lena



(b) Boat

Figura 5. Plots

Referencias

[1] Jared Tanner y Ke Wei. «Low rank matrix completion by alternating steepest descent methods». En: *Applied and Computational Harmonic Analysis* 40.2 (2016), págs. 417-429. ISSN: 1063-5203. DOI: https://doi.org/10.1016/j.acha.2015.08.003. URL: http://www.sciencedirect.com/science/article/pii/S1063520315001062.