Análisis de la velocidad del viento en Irlanda

Autores: Alejandro Magnorsky, Andrés Mata Suárez, Mariano Merchante

Instituto Tecnológico de Buenos Aires

Resumen

Palabras clave

Velocidad del viento; Cuadrados mínimos; Irlanda.

I. Introducción

La energía eólica es aquella obtenida a partir del viento. Se trata de energía cinética generada por medio de las corrientes de aire, la cual es transformada en otras formas útiles de energía para las actividades humanas. Las ventajas de la energía eólica es que es un recurso abundante, renovable y limpio. Sin embargo, el principal inconveniente es su intermitencia.

Para poder aprovechar la energía eólica es importante conocer la velocidad del viento. Esto incluye la velocidad máxima y mínima. Para poder utilizar la energía del viento, es necesario que éste alcance una velocidad mínima que depende del aerogenerador que se vaya a utilizar pero que suele empezar entre los 3 m/s y los 4 m/s, sin exceder los 25 m/s. En los aerogeneradores la energía eólica mueve una hélice y mediante un sistema mecánico se hace girar el rotor de un generador que produce energía eléctrica. Para que su instalación resulte rentable, suelen agruparse en concentraciones denominadas parques eólicos.

El gobierno irlandés consideró la posibilidad de utilizar la energía eólica para satisfacer una porción significativa de las necesidades energéticas de Irlanda. Para determinar la conveniencia y los lugares con mayor potencial se realizó un relevamiento de las velocidades del viento medidas por 12 estaciones meteorológicas distribuidas a lo largo del territorio. Cabe mencionar que existen distintas escalas de implementación de energía eólica: desde la calefacción de edificios hasta el abastecimiento de energía eléctrica de un cierto porcentaje de la población de una ciudad.

Antes de instalar y comenzar a operar un aerogenerador en un potencial lugar para producir otras fuentes de energía usando la eólica se suele utilizar registros de las velocidades del viento en dicha zona a lo largo de varios meses. La energía que puede producir un aerogenerador es una función no lineal de la velocidad del viento. Por lo tanto, el cálculo de la velocidad media del viento en la zona es insuficiente. Se hace, entonces, necesaria la estimación de la distribución completa de las velocidades (Haslett, J. y Raftery, A. E., 1989).

En la sección II se explica de que manera se ajustan los datos de las velocidades a una función usando cuadrados mínimos y cómo se calcula el error cuadrático medio. En la sección III se exponen, para cada estación meteorológica, las funciones que se ajustan a los datos, el error cuadrático medio y las velocidades medias del viento. En la sección IV se sacan comparan los resultados y se mencionan las características más importantes encontradas. En el apéndice se detallan secciones del código utilizado para resolver el problema de cuadrados mínimos.

II. Desarrollo

A. Aproximación de los datos a una función

Para cada una de las estaciones meteorológicas, se desea ajustar las velocidades medidas a una función de la forma:

$$v(t) = A_0 + A_1 cos(2\pi f_1 t) + B_1 sen(2\pi f_1 t) \tag{1}$$

donde t es el tiempo en días, $v(t)$ es la velocidad del viento en el instante t y $f_1 = \frac{1}{365{,}25} dia^{-1}$.
$v(t)$ es lineal en función de A_0, A_1 y B_1 y se puede expresar como $v(t) = A_0 f_1(t) + A_1 f_2(t) + B_1 f_3(t)$ donde $f_1(t) = 1$, $f_2(t) = cos(2\pi f_1 t)$ y $f_3(t) = sen(2\pi f_1 t)$.

Para resolver el problema de ajuste se utiliza el método de cuadrados mínimos (Mathews y Fink, 1992). El objetivo entonces es encontrar A_0, A_1 y B_1 tal que $\sum_{k=1}^{n} (v_k - v(t_k))^2$ sea mínimo.

Estación meteorológica	Velocidad media (m/s)
Roche's Pt.	6.3604
Valentia	5.4770
Rosslare	5.9984
Kilkenny	3.2442
Shannon	5.3794
Birr	3.6485
Dublin	5.0399
Claremorris	4.3699
Mullingar	4.3706
Clones	4.4794
Belmullet	6.7500
Malin head	8.0250

Tabla I: Velocidades medias en cada estación meteorológica.

Si definimos la matriz A y los vectores $\vec{x}$ y $\vec{b}$ como:

$$A = \begin{pmatrix} f_1(t_1) & f_2(t_1) & f_3(t_1) \\ f_1(t_2) & f_2(t_2) & f_3(t_2) \\ \vdots & \vdots & \vdots \\ f_1(t_n) & f_2(t_n) & f_3(t_n) \end{pmatrix} \qquad \vec{x} = \begin{pmatrix} A_0 \\ A_1 \\ B_1 \end{pmatrix} \qquad \vec{b} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} \tag{2}$$

entonces, el objetivo puede expresarse como encontrar $\vec{x}$ tal que $||A\vec{x} - b||^2$ sea mínimo.

La solución clásica al problema de cuadrados mínimos es hacer $\vec{x} = (A^T A)^{-1} A^T \vec{b}$. Sin embargo, calcular la inversa de una matriz es de $O(n^3)$. Es por eso que, en general, se opta por resolver el problema utilizando la factorización QR obtenida por:

- Householder
- Gram-Schmidt
- Givens

La forma de resolver el problema de cuadrados mínimos usando QR consiste en:

$$A = QR = (Q_1\, Q_2) \begin{pmatrix} R_1 \\ 0 \end{pmatrix} \tag{3}$$

donde A tiene dimensiones $nx3$, Q_1 de $nx3$, Q_2 de $nx(n-3)$ y R_1 de $3x3$.

La solución se obtiene resolviendo:

$$R_1 \vec{x} = Q_1^T \vec{b} \tag{4}$$

por sustitución hacia atrás ya que R_1 es triangular superior.

B. Error cuadrático medio

El error cuadrático medio se calcula mediante la siguiente ecuación:

$$E_{cm} = \sqrt{\frac{\sum_{k=1}^{n} e_k^2}{n}} = \sqrt{\frac{||Q_2^T \vec{b}||^2}{n}} \tag{5}$$

siendo e_k el error en un punto (t_k, v_k), tal que:

$$e_k = v_k - v(t_k) \tag{6}$$

para un conjunto de n mediciones de la velocidad el viento v_k para el día número t_k.

III. Resultados

A. Cálculo de las velocidades medias

Con el fin de conocer mejor cómo son los datos obtenidos de la medición de la velocidad del viento en las distintas estaciones, se calcula el promedio de todos los datos en cada una de ellas. El resultado de ello se encuentra en la tabla I.

Estación meteorológica	A_0 (m/s)	A_1 (m/s)	B_1 (m/s)
Roche's Pt.	6.360473	1.098654	0.273242
Valentia	5.477037	1.031857	0.228245
Rosslare	5.998487	0.828876	0.372581
Kilkenny	3.244228	0.367452	0.327301
Shannon	5.379480	0.560103	0.308363
Birr	3.648573	0.447176	0.239542
Dublin	5.039962	1.030374	0.370597
Claremorris	4.369907	0.489844	0.342406
Mullingar	4.370628	0.512658	0.312935
Clones	4.479414	0.620200	0.368142
Belmullet	6.750024	0.700321	0.074164
Malin head	8.025106	1.568598	0.024888

Tabla II: Valores de los coeficientes A_0, A_1 y B_1 para cada estación meteorológica.

Estación meteorológica	E_{cm} (m/s)
Roche's Pt.	2.7777
Valentia	2.6051
Rosslare	2.4946
Kilkenny	1.8217
Shannon	2.4984
Birr	2.0097
Dublin	2.4404
Claremorris	2.2754
Mullingar	2.1014
Clones	2.2598
Belmullet	2.9600
Malin head	3.2619

Tabla III: Error cuadrático medio para cada estación meteorológica.

B. Cálculo de las funciones que se ajustan a los datos

Utilizando la ecuación 4 se obtuvieron los coeficientes que figuran en la tabla II.

C. Cálculo de los errores cuadráticos medios

La tabla III muestra los resultados del cálculo del error cuadrático medio para cada estación meteorológica considerada.

D. Histogramas de los errores

IV. Conclusiones

A. Valor medio de días por año

Si se realiza un promedio P de la cantidad de días de cómputo, comprendidos entre los años 1961 y 1978, durante los cuales se estudió la velocidad del viento en cada estación, se obtiene:

$$P = \frac{\sum_{i=1}^{1978-1961+1} cantidad\,de\,dias\,del\,año\,i}{1978-1961+1} = 365.22 \approx 365.25 \tag{7}$$

Considerando la función de aproximación $v(t)$, más precisamente las funciones sinusoidales $f2(t)$ y $f3(t)$, podemos ver que se está utilizando este valor P como período de oscilación; en otras palabras, se estima un intervalo de tiempo dentro del cual la velocidad del viento variará tanto como tenga que variar, volviendo a su valor inicial y repitiendo tal proceso cada 365.25 días.

Se elige este promedio como período de oscilación debido a la existencia de años bisietos (un año de 366 días cada 4 años). Esto implica que, al considerarse un promedio de 365 días por año, cada 4 años se estaría produciendo un error de un día. Particularmente, para el caso analizado en estas hojas se tiene que los años 1964, 1968, 1972 y 1976 son bisiestos.

B. El valor del coeficiente A_0

Como la muestra de datos se desarrolló durante un período de 18 años completos, si analizamos la ecuación 1 podemos ver que tanto el segundo término como el tercero se pueden despreciar, ya que

$$\int_0^{(1978-1961+1)365.25=6574.5} sin(\frac{2\pi t}{365.25}) = \frac{1 - cos(\frac{2\pi 6574.5}{365.25})}{\frac{2\pi}{365.25}} = 0 \tag{8}$$

$$\int_0^{6574.5} cos(\frac{2\pi t}{365.25}) = \frac{sin(\frac{2\pi 6574.5}{365.25})}{\frac{2\pi}{365.25}} = 0 \tag{9}$$

por lo que el valor A_0 siempre tenderá al valor medio de la velocidad del viento en cada estación metereológica.

REFERENCIAS

Haslett, J., Raftery, Adrian E., "Space-time Modelling with Long-memory Dependence: Assessing Ireland's Wind Power Resource", 1989

Mathews, John H., Fink, Kurtis D., "Numerical Methods Using MATLAB", Prentice Hall, 1999

V. Apéndice

A. Detalles sobre los algoritmos para la factorización QR

A continuación se detallan de manera general algunos puntos considerados relevantes para cada una de las implementaciones de los métodos de descomposición QR considerados en este trabajo, para una matriz A de mxn, con $m \geq n$.

1) Reflexiones de Householder: El algoritmo consiste en usar las matrices de Householder para hacer 0 todos los elementos debajo de la diagonal para cada columna de la matriz A. La matriz de Householder se define como:

$$H = \mathbb{I} - 2\frac{\vec{u}\vec{u}^T}{||\vec{u}||^2} \tag{10}$$

con $\vec{u} = \vec{x}$ excepto en el elemento que no queremos que sea 0, donde toma el valor $\vec{u}_k = \vec{x}_k + \sigma$, siendo $\vec{x}$ el vector columna que sobre el que se está operando y $\sigma = sign(x_k)||\vec{x}||$.

Con el fin de optimizar el algoritmo se realizaron ciertas modificaciones usando las siguientes igualdades:

$$||\vec{u}||^2 = \sum_{i=1}^{n} u_i^2 = \sum_{i=2}^{n} x_i^2 + (x_1 + \sigma)^2 = \sum_{i=2}^{n} x_i^2 + x_1^2 + 2x_1\sigma + \sigma^2 = 2(||\vec{x}||^2 + x_1\sigma) \tag{11}$$

$$R_{k:m,k} = H\vec{x} = \vec{x} - 2\frac{\vec{u}\vec{u}^T\vec{x}}{||\vec{u}||^2} = \vec{x} - \vec{u}\frac{2(||\vec{x}||^2 + x_1\sigma)}{||\vec{u}||^2} = \vec{x} - \vec{u} \tag{12}$$

$$R_{k,k} = \vec{x}_1 - \vec{u}_1 = \vec{x}_1 - (\vec{x}_1 + \sigma) = -\sigma \tag{13}$$

A su vez, como el cálculo de Q es muy costoso ya que requiere de la multiplicación de matrices de mxm (m es el número de filas), se optó por calcular directamente $Q_1\vec{b}$, ya que para ello se multiplican entre sí vectores y no matrices. La desventaja de esto es que el algoritmo devuelve los coeficientes de la aproximación, en lugar de Q y R. Esto implica que hay que ejecutarlo una vez para cada función que se quiera aproximar a los datos. En cambio, el algoritmo de Givens y el de Gram-Schmidt se ejecutan una sola vez ya que devuelven las matrices Q y R que luego se utilizan para realizar sustitución hacia atras y obtener los coeficientes para los distintos valores de $\vec{b}$.

2) Método de ortogonalización de Gram-Schmidt: Para la descomposición QR, el método de ortogonalización de Gram-Schmidt puede utilizarse sobre las columnas de la matriz $A = [a_1, \cdots, a_n]$ donde cada vector a_i tiene m elementos. Para la ortogonalización, se calculan los vectores u_k y e_k de forma tal que

$$u_k = a_k - \sum_{j=1}^{k-1} proj_{e_j} a_k \tag{14}$$

$$e_k = \frac{u_k}{\parallel u_k \parallel} \tag{15}$$

De esta forma, y considerando que todos los vectores e_k son vectores unitarios y forman una base ortogonal debido a su construcción, se pueden definir $Q = [e_1, \cdots, e_n]$ ortogonal y R una matriz triangular superior tal que

$$\begin{bmatrix} \langle e_1, a_1\rangle & \langle e_1, a_2\rangle & \cdots & \langle e_1, a_n\rangle \\ 0 & \langle e_2, a_2\rangle & \cdots & \langle e_2, a_n\rangle \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \langle e_m, a_n\rangle \end{bmatrix} \tag{16}$$

La implementación del algoritmo es considerablemente directa con buenos resultados; sin embargo en términos de estabilidad numérica el algoritmo es inestable debido a errores de redondeo por punto flotante. Por otro lado, una pequeña optimización utilizada fue aprovechar el cálculo de los vectores e_k para construir la matriz R progresivamente.

3) Rotaciones de Givens: La idea basamental del algoritmo es ir anulando, mediante rotaciones de Givens, los elementos de la matriz A que se encuentren por debajo de la diagonal de manera ascendente (de abajo hacia arriba, de izquierda a derecha), empezando por el elemento en la posición $(m, 1)$, de manera tal que en la última iteración A se convierta en la matriz triangular superior R de la descomposición.

En cada paso del algoritmo, se busca rotar el elemento (i, j) con el elemento $(i-1, j)$; esto provoca la anulación del primero. Para dicho cometido, se utiliza una matriz de rotación de Givens $G = G(i)$, que se encarga de rotar tales elementos un ángulo $\theta = arctan(\frac{-A(i,j)}{A(i-1,j)})$. La anulación del elemento (i, j) se produce mediante el producto de la matriz de rotación y la matriz A.

Definamos entonces a $G_k = G_k(i)$ como la matriz de rotación de Givens para los elementos (i, j) y $(i-1, j)$ en el paso k del algoritmo, a $A_{k+1} = G_{k+1}A_k$ como el estado de la matriz inicial A en la $(k+1)$-ésima iteración y a s como la cantidad de iteraciones del algoritmo. Se tiene, por lo tanto:

$$R = (\prod_{t=1}^{s} G_{s-t+1})A \tag{17}$$

Podemos suponer que se utilizarán tantas matrices de rotación como elementos no nulos haya debajo de la diagonal de A, una en cada iteración. De manera análoga, definimos $Q_{k+1} = Q_k * G_{k+1}^T$ como el estado de la matriz Q de la descomposicón en la $(k+1)$-ésima iteración. Finalmente,

$$Q = \prod_{t=1}^{s} G_t^T \tag{18}$$

Debido a las características de las matrices de rotación de Givens (valores nulos excepto en los elementos de las diagonales y las posiciones $(i, i-1)$ y $(i-1, i)$), el producto A_{k+1} puede ser computado mediante la modificación de las filas i e $(i-1)$-ésimas de A_k únicamente. Lo mismo ocurre si hablamos de Q_{k+1}: es necesario alterar solamente las columnas i e $(i-1)$-ésimas de Q_k. Es aquí dónde el algoritmo de Givens obtiene un gran punto a favor: no es necesario calcular el producto matricial entero; sólo basta realizar los cambios necesario en las filas o columnas mencionadas. Esto es especialmente ventajoso en casos en dónde se esté trabajando con matrices de dimensiones superlativas, en donde la memoria disponible para las operaciones se vuelve un factor delicado, teniendo en cuenta, además, que ambos productos deben realizarse en cada iteración.

Listing 1: Implementación de la factorización QR usando reflexiones de Householder.

```
% A es una matriz de mxn con m >= n
% b es un vector de mx1
% Devuelve el vector x de nx1 tal que R1*x = Q1'*b
function [x,e] = householder(A,b)
        [m, n] = size(A);
        R = A;

        for k=1:n
                x = R(k:m,k);
                % ||x||^2 = <x,x> = x'*x
                nx2 = x'*x;
                sigma = sign(x(1))*sqrt(nx2);
                u = x;
                u(1) = u(1)+sigma;

                H = eye(m-(k-1)) - (1/(nx2+x(1)*sigma)).*(u*u');

                R(k,k) = -sigma;
                R(k+1:m,k) = 0;
                if k < n
                        R(k:m,k+1:n) = H*R(k:m,k+1:n);
                endif

                % Calcula Q1'*b en lugar de calcular Q
                b(k:m) = H*b(k:m);
        endfor

        % Calcula x por sustitucion hacia atras
        x = zeros(n,1);
        for k=n:-1:1
                x(k) = (b(k) - R(k,k+1:n)*x(k+1:n))/R(k,k);
        end

        % Calcula el error minimo
        e = b(n+1:m)'*b(n+1:m);
endfunction
```

Listing 2: Implementación de la factorización QR por medio del método de Gram-Schmidt.

```
% A es una matriz de mxn con m >= n
% Devuelve las matrices Q y R
 function [Q,R] = gramSchmidt(A)
        [m, n] = size(A);
        Q = zeros(m,n);
        R = zeros(m,n);
        e = zeros(m,n);

        for k=1:n
                accum = zeros(m,1);
                for j=1:k-1
                        accum = accum + e(:,j)  .*      (  (e(:,j)'*A(:,k))   /   (e(:,j)'*e(:,j))  );
                endfor
                u = A(:,k) - accum;
                e(:,k) = u / norm(u);

                for j=k:n
                        R(k,j) = (e(:,k)'*A(:,j));
                endfor
        endfor

        Q = e;
endfunction
```

Listing 3: Implementación de la factorización QR por medio rotaciones de Givens.

```
% A es una matriz de mxn con m >= n
% Devuelve las matrices Q y R
function [Q R] = givens(A)
        m = length(A);
        n = length(A(1,:));

        Q = eye(m);
        R = A;
        aux = zeros(m);

        % Rotando y anulando de abajo hacia arriba, de izquierda a derecha.
        % Ejemplo para una matriz A de 4x4:
        %       X X X X         X X X X         X X X X         X X X X
        %       X X X X         X X X X         X X X X         0 X X X
        %  -->  X X X X   -->   X X X X   -->   0 X X X   -->   0 X X X   -->
        %       X X X X         0 X X X         0 X X X         0 X X X
        %       se rota (4,1)   se rota (3,1)   se rota (2,1)   se rota (4,2)
        %       con (3,1)       con (2,1)       con (1,1)       con (3,2)
        %
        %       X X X X         X X X X         X X X X
        %       0 X X X         0 X X X         0 X X X
        %  -->  0 X X X   -->   0 0 X X   -->   0 0 X X
        %       0 0 X X         0 0 X X         0 0 0 X
        %       se rota (3,2)   se rota (4,3)   triangular
        %       con (2,2)       con (3,3)       superior

        for j = 1:n;
                for i = m:-1:j+1
                        if (R(i,j) != 0)
                                a = R(i-1,j);           % Se rotara b con a
                                b = R(i,j);             % b sera el que resulte anulado
                                d = sqrt(a^2 + b^2);    % distancia ab en el plano xy
                                c = a / d;              % coseno
                                s = -b / d;             % seno

                                % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                %  La rotacion del elemento en (i,j) con el elemento
                                % en (i-1,j) solo afecta a las filas i e i-1 de R.
                                %  Es ineficiente realizar un producto matricial
                                % entero si solo son afectadas algunas filas,
                                % especialmente en matrices de dimensiones grandes.
                                % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                                %  Las siguientes asignaciones mejoran el producto:
                                %       R = G * R
                                %  donde:
                                %       - G es la matriz de rotacion de Givens para
                                % rotar (i,j) con (i-1,j).
                                %       - R es la matriz triangular superior parcial,
                                % para un cierto paso del algoritmo.

                                aux = [R(i-1,:); R(i,:)];
                                R(i-1,:) = c * aux(1,:) - s * aux(2,:);
                                R(i,:) = s * aux(1,:) + c * aux(2,:);

                                %  Las siguientes asignaciones mejoran el producto:
                                %       Q = Q * G'
                                %  donde:
                                %       - G' es la matriz de rotacion de Givens traspuesta
                                % para rotar (i,j) con (i-1,j).
                                %       - Q es la matriz ortogonal parcial de la descomposicion,
                                % para un cierto paso del algoritmo.

                                aux = [Q(:,i-1)  Q(:,i)];
                                Q(:,i-1) = c * aux(:,1) - s * aux(:,2);
                                Q(:,i) = s * aux(:,1) + c * aux(:,2);
                        endif
                endfor
        endfor
endfunction
```

Listing 4: Implementación de la resolución del sistema por el método de cuadrados mínimos.

```
function x = prob3(qrFunction)

        data = load("../data/windms.data");
        n = length(data(:,1));
        data = data(:,4:end);

        t = 1:n;
        f1 = 1/365.25;
        A(1:n,1) = 1;
        A(1:n,2) = cos(2*pi*f1*t);
        A(1:n,3) = sin(2*pi*f1*t);
        [Q,R] = qrFunction(A);
        R1 = R(1:3,:);
        Q1 = Q(1:n,1:3);

        for k=1:12
                b = data(:,k);
                x(:,k) = sust(R1, Q1, b);
        endfor
endfunction
```

Listing 5: Implementación de la sustitución hacia atrás.

```
function x = sust(R1,Q1,b)

n = length(R1(:,1));
b = Q1'*b;
x = zeros(n,1);

for k=n:-1:1
        x(k) = (b(k) - R1(k,k+1:n)*x(k+1:n))/R1(k,k);
end
```