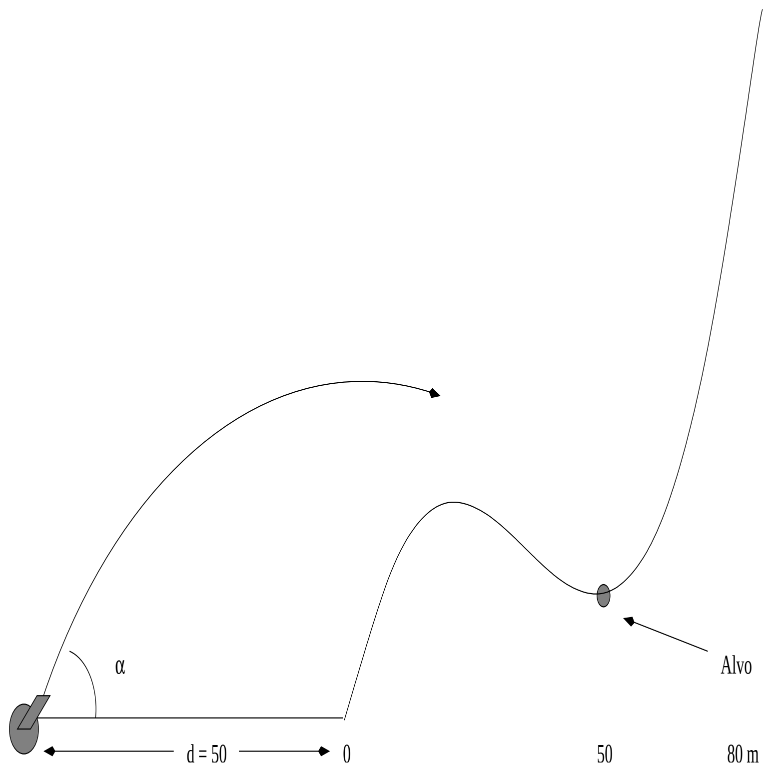
MAP-3121 - Primeira tarefa computacional - 2015

Consideremos um canhão que fará um disparo visando atingir um alvo localizado atrás de uma montanha. O canhão se encontra em uma região plana a uma distância $d = 50m$ do sopé da montanha. O alvo se encontra num vale atrás da primeira encosta do morro a 50m (em linha reta) do sopé da montanha (conforme a figura). Para atingirmos o alvo podemos ajustar o ângulo do disparo.



Equacionemos o problema, tomando o sopé da montanha como sendo o ponto$(x, y) = (0, 0)$. Considere que o canhão imprime uma velocidade inicial de $50+r$ m/s à bala, onde r é igual ao número formado pelos dois últimos digitos de seu número USP dividido por 10. Assim se seu número USP fosse 1234567, você deveria adotar a velocidade inicial de 56.7 m/s. A altura da encosta do morro (em metros) é dada pela equação (para $x \geq 0$): $y = (x^3 - 105x^2 + 3000x)/1000$. A equação do movimento da bala do canhão disparado a uma distância d da base do morro e com ângulo α de disparo, é dada por: $x(t) = -d+(50+r)t\cos\alpha$ e $y(t) = (50 + r)t\sin\alpha - gt^2/2$. (Tome $g = 10m/s^2$) Usando o valor de t da primeira equação na segunda obtemos a equação para a trajetória da bala, em função de α. O alvo está localizado na superfície da encosta, no ponto $x = 50$

(portanto a uma altura 12.5).

Em seu exercício você deverá ter um procedimento que dado α calcule qual ponto da montanha é atingido pelo disparo do canhão e deverá determinar para qual (ou quais) valores de α o alvo é atingido. Para tal você precisará avaliar o valor de polinômios e talvez de suas derivadas. Ao final descrevemos um algoritmo (de Horner) que é eficiente para fazer isto.

Entrega da tarefa

Sua tarefa deve ser entregue no sistema Moodle (no grauna.ime.usp.br) até o dia 22/03. Você entregará o fonte do programa (escrito em C ou Python) e mais um relatório (em PDF) de no máximo duas páginas descrevendo suscintamente seu método de solução do problema e os resultados, com gráficos da trajetória da bala para atingir o alvo. Divirta-se! PS - Esta tarefa deve ser feita individualmente!

O algoritmo de Horner - Pense em como você implementaria um algoritmo para calcular o valor de um polinômio de grau n em um ponto x qualquer. Quantas multiplicações e adições seriam necessárias? O algoritmo de Horner avalia o valor de um polinômio com um número mínimo de operações. Na avaliação de $p(x) = a_0x^n + a_1x^{n-1} + ... + a_{n-1}x + a_n$ em um dado ponto z utiliza-se a fatoração:

$$p(z) = ((...((a_0z + a_1)z + a_2)z + ...)\, z + a_{n-1})\, z + a_n$$

Dado z e os coeficientes a_i de p, o valor de $p(z)$ é computado através dos passos:

$b_0 = a_0$

Para $i = 1$ até n faça

$b_i = b_{i-1} * z + a_i$

O valor de $p(z)$ é dado por b_n. No método de Newton necessitamos não apenas do valor do polinômio mas também o de sua derivada. Para tal considere o polinômio

$$q(x) = b_0x^{n-1} + b_1x^{n-2} + ... + b_{n-2}x + b_{n-1} \quad ,$$

com os coeficientes b_i gerados no algoritmo de Horner. Temos a seguinte relação:

$$\begin{aligned}
(x - z)q(x) &= (x - z)(b_0x^{n-1} + b_1x^{n-2} + ... + b_{n-2}x + b_{n-1}) \\
&= b_0x^n + (b_1 - zb_0)x^{n-1} + ... + (b_{n-1} - zb_{n-2})x + (b_n - zb_{n-1}) - b_n \\
&= a_0x^n + a_1x^{n-1} + a_2x^{n-2} + ... + a_{n-1}x + a_n - p(z) \\
&= p(x) - p(z)
\end{aligned}$$

onde usamos a definição dos coeficientes b_i em função dos coeficientes de $p(x)$. Da igualdade $p(x) = p(z) + (x - z)q(x)$ obtida acima, derivando com respeito a x, chegamos à expressão para a derivada de p(x):

$$p'(x) = (x - z)q'(x) + q(x)$$

Se tomarmos $x = z$ nesta expressão obtemos que $p'(z) = q(z)$. Assim podemos avaliar o valor da derivada de p no ponto z calculando o valor de $q(z)$, o que pode ser feito através do algoritmo de Horner (lembrando que q tem grau $n-1$). Ou seja, calculamos:

$c_0 = b_0$

Para $i = 1$ até $n - 1$ faça

$c_i = c_{i-1} * z + b_i$

Obtemos como anteriormente que $q(z) = c_{n-1}$ e portanto $p'(z) = c_{n-1}$. Temos portanto todos os ingredientes necessários para obter de forma eficiente a cada passo do método de Newton os valores do polinômio e sua derivada.

Um pouco mais sobre o algoritmo de Horner

Da mesma forma como procedemos anteriormente podemos voltar a aplicar o algoritmo, agora para o cálculo de $r(z)$, onde definimos

$$r(x) = c_0 x^{n-2} + c_1 x^{n-3} + ... + c_{n-3} x + c_{n-2} \quad ,$$

utilizando os coeficientes c_i obtidos através do algoritmo. Calculamos então:

$d_0 = c_0$

Para $i = 1$ até $n - 2$ faça

$d_i = d_{i-1} * z + c_i$

Assim obtemos que $r(z) = d_{n-2}$ e podemos deduzir a relação:

$$q(x) = (x - z)r(x) + q(z)$$

da mesma forma como obtivemos a relação entre os polinômios p e q, dada por $p(x) = (x - z)q(x) + p(z)$. Derivando esta última expressão duas vezes obtemos

$$\begin{aligned} p''(x) &= (x - z)q''(x) + 2q'(x) \\ &= (x - z)q''(x) + 2(x - z)r'(x) + 2r(x) \end{aligned}$$

Na última igualdade usamos que $q'(x) = (x - z)r'(x) + r(x)$. Tomando $x = z$ obtemos que $p''(z)/2 = r(z) = d_{n-2}$. Podemos continuar aplicando o algoritmo, cada vez a um polinômio de grau 1 a menos, obtido do anterior. Para descrevermos todo este processo vamos no entanto introduzir uma nova notação, pois vamos necessitar de dois índices (vamos usar a variável $m_{i,j}$). Definimos inicialmente para $j = -1$ os valores $m_{i,-1} = a_i$, $i = 0, 1, ..., n$, através dos coeficientes de $p(x)$. O algoritmo de Horner completo é dado por:

Para $j = 0$ até n faça

$m_{0,j} = m_{0,j-1}$

Para $i = 1$ até $n - j$ faça

$$m_{i,j} = m_{i-1,j} * z + m_{i,j-1}$$

Ao final do algoritmo temos que $p^{(k)}(z)/k! = m_{n-k,k}$. (Note que na notação anterior b_n corresponde a $m_{n,0}$, c_{n-1} a $m_{n-1,1}$ e d_{n-2} a $m_{n-2,2}$.) O algoritmo quando executado até o final fornece todos os coeficientes do polinômio de Taylor do polinômio p em torno do ponto z. É fácil ver então que:

$$p(x) = m_{n,0} + m_{n-1,1}(x - z) + m_{n-2,2}(x - z)^2 + ... + m_{0,n}(x - z)^n$$

uma vez que um polinômio de grau n tem que ser idêntico a seu polinômio de Taylor de grau maior ou igual a n.

Exemplo - Considere o polinômio $p(x) = 2x^4 - x^3 + 3x^2 - 2x + 5$. Mostramos a seguir os resultados obtidos para $z = 2$ através da execução do algoritmo.

$j = -1$	2	-1	3	-2	5
$j = 0$	2	3	9	16	37
$j = 1$	2	7	23	62	
$j = 2$	2	11	45		
$j = 3$	2	15			
$j = 4$	2				

Como resultados temos que $p(2) = 37$ e $p'(2) = 62$. Além disso podemos escrever $p(x)$ na forma de seu polinômio de Taylor como:

$$p(x) = 37 + 62(x - 2) + 45(x - 2)^2 + 15(x - 2)^3 + 2(x - 2)^4$$

Verifique! Repita o mesmo para o polinômio $p(x) = x^4 - 2x^3 + 3x^2 + 3x - 7$ no ponto $z = 3$.

Observação - Note que a implementação do Algoritmo de Horner completo na verdade não requer o uso da matriz bidimensional $m_{i,j}$. Esta poderia ser feita usando apenas um vetor onde se armazena uma cópia dos coeficentes de p e sobrescrevendo suas posições a cada passo. Ao final do processo o vetor não mais conteria os coeficientes de p, mas sim seus coeficentes do polinômio de Taylor. Em outras palavras, este algoritmo, a partir dos coeficientes de p na base usual $\{1, x, .., x^n\}$, gera seus coeficientes na base $\{1, (x - z), ..., (x - z)^n\}$. Portanto o algoritmo de Horner pode ser visto como um algoritmo de mudança de base!