1 Introduzione: matematica

1.1 Derivate numeriche

Per prima cosa inizio con un piccolo elenco di derivate numeriche, usando il metodo delle differenze numeriche:

La derivata prima (in avanti) è:

$$\frac{\partial F}{\partial x}(a) \simeq \frac{F(a+h) - F(a)}{h} + O(h) \tag{1.1.1}$$

Ma la sua precisione è al primo ordine, per cui utilizzerò la versione cosiddetta "centrale":

$$\frac{\partial F}{\partial x}(a) \simeq \frac{F(a+h) - F(a-h)}{2 * h} + O(h^2) \tag{1.1.2}$$

Per la derivata seconda il discorso e' simile ma in entrambi i casi la precisione è sempre al secondo ordine:

$$\frac{\partial^2 F}{\partial x^2}(a) \simeq \frac{\frac{\partial F}{\partial x}(a+h) - \frac{\partial F}{\partial x}(a)}{h} = \frac{F(a+2h) + F(a) - 2F(a+h)}{h^2} + O(h^2) \tag{1.1.3}$$

$$\frac{\partial^2 F}{\partial x^2}(a) \simeq \frac{F(a+h) + F(a-h) - 2F(a)}{h^2} + O(h^2) \tag{1.1.4}$$

1.2 Equazione del calore

Per prima cosa parto dall'equazione del calore

$$\frac{\partial T}{\partial t} = K \frac{\partial^2 T}{\partial x^2} \tag{1.2.1}$$

Per calcolare l'equazione come prima cosa calcoliamo le derivate:

$$\frac{\partial T}{\partial t}(x,t) = \frac{T(x, t+\Delta t) - T(x,t)}{\Delta t} \tag{1.2.2}$$

e

$$\frac{\partial^2 T}{\partial x^2}(x,t) = \frac{T(x+\Delta x, t) + T(x-\Delta x, t) - 2T(x,t)}{\Delta x^2} \tag{1.2.3}$$

Per prima cosa eguaglio le derivate (ma devo ricordare che la risoluzione che si avrebbe precisione Δx^2 nello spazio e Δt nel tempo):

$$T(x, t+\Delta t) = k \frac{\Delta t}{\Delta x^2} \left(T(x+\Delta x, t) + T(x-\Delta x, t) - 2T(x,t)\right) + T(x,t) \tag{1.2.4}$$

e andando avanti:

$$\frac{T(x, t+\Delta t) - T(x,t)}{\Delta t} = k \frac{T(x+\Delta x, t) + T(x-\Delta x, t) - 2T(x,t)}{\Delta x^2} \tag{1.2.5}$$

A questo punto calcolo ogni punto conoscendo i tre del passo precedente, ma non aproffondirò la cosa.

Oppure posso mettere ugualiare con la derivata spaziale all'istante sucessivo:

$$\frac{T(x, t+\Delta t) - T(x,t)}{\Delta t} = k \frac{T(x+\Delta x, t+\Delta t) + T(x-\Delta x, t+\Delta t) - 2T(x, t+\Delta t)}{\Delta x^2} \tag{1.2.6}$$

proseguo:

$$T(x, t+\Delta t) - k \frac{\Delta t}{\Delta x^2} \left(T(x+\Delta x, t+\Delta t) + T(x-\Delta x, t+\Delta t) - 2T(x, t+\Delta t)\right) = T(x,t) \tag{1.2.7}$$

Anche in questo caso non approfondisco, preferisco ricavare il metodo di Crank-Nicolson per poi generalizzarlo e descriverne l'algoritmo utilizzato per la risoluzione dell'equazione:

1.3 Un'esempio di come ricavare il metodo di Crank Nicolson: l'equazione del calore

Innanzitutto utilizzamo la definizione centrale della derivata prima in modo da avere una precisione di Δt^2 anche per quanto riguarda il tempo, ma la calcolo in $t + \Delta t/2$ con incremento Δt:

$$\frac{\partial T}{\partial t}(x, t + \Delta t/2) \simeq \frac{T(x, t + \Delta t/2 + \Delta t/2) - T(x, t - \Delta t/2 + \Delta t/2)}{2 * \Delta t/2} = \frac{T(x, t + \Delta t) - T(x, t)}{\Delta t} \tag{1.3.1}$$

Per lo spazio utilizziamo le derivate calcolate negli esempi precedenti. A questo punto abbiamo $\frac{\partial T}{\partial t}(x, t + \Delta t/2)$, $\frac{\partial^2 T}{\partial x^2}(x, t)$ e $\frac{\partial^2 T}{\partial x^2}(x, t + \Delta t)$ per rispettare l'equazione facciamo la media delle due derivate spaziali ai tempi t e $t + \Delta t$.

$$\begin{aligned}\frac{T(x, t + \Delta t) - T(x, t)}{\Delta t} = \frac{K}{2} \Bigg(& \frac{T(x + \Delta x, t) + T(x - \Delta x, t) - 2T(x, t)}{\Delta x^2} \\ & + \frac{T(x + \Delta x, t + \Delta t) + T(x - \Delta x, t + \Delta t) - 2T(x, t + \Delta t)}{\Delta x^2} \Bigg)\end{aligned} \tag{1.3.2}$$

In pochi passaggi si arriva a separare le parti a tempo differente:, con $\eta = K \frac{\Delta t}{\Delta x^2}$:

$$\left(\frac{2}{\eta} + 2\right) T(x, t + \Delta t) - T(x + \Delta x, t + \Delta t) - T(x - \Delta x, t + \Delta t) = \left(\frac{2}{\eta} - 2\right) T(x, t) + T(x + \Delta x, t) + T(x - \Delta x, t) \tag{1.3.3}$$

Per ogni istante di tempo ho una matrice tridiagonale per il passo temporale che conosco e per quello successivo. In sottosezione 2.2 descriverò come si risolve una di queste matrici.

2 Costruzione dell'algoritmo

2.1 Ottenere il sistema

Ho fatto un esempio col l'equazione del calore. Prima di procedere alla spiegazione su come si semplifica e si risolve un sistema di equazioni riconducibile ad una matrice tridiagonale spiegherò come condurre la PDE più generale ad un sistema del genere.

Partendo da una generica equazione di secondo ordine:

$$\partial_t F = D_2 \partial_x^2 F + D_1 \partial_x F + V(x,t)F + U(x,t) \tag{2.1.1}$$

Il coefficiente della derivata temporale è ignorato perché è incluso negli altri coefficienti, ovviamente D_2, il coefficiente della derivata seconda spaziale non deve essere mai uguale a zero! Inoltre preferisco lasciare D_1 e D_2 come costanti nel tempo e nello spazio, e lascio la dipendenza temporale e spaziale a V e U, che possono essere costanti a loro volta.

In seguito indicherò la il passo nello spazio a pedice con i e quello nel tempo in apice con j. Discretizzando l'equazione ottengo quindi (ho mantenuto la definizione di derivata centrale anche per quella prima spaziale in modo da mantenere la precisione).

$$\begin{aligned}\frac{F_i^{j+1} - F_i^j}{\Delta t} = \frac{1}{2}\Bigg(& D_2 \frac{F_{i+1}^j + F_{i-1}^j - 2F_i^j}{\Delta x^2} + D_1 \frac{F_{i+1}^j - F_{i-1}^j}{2\Delta x} + F_i^j V_i^{j+1/2} + U_i^{j+1/2} + \\ & D_2 \frac{F_{i+1}^{j+1} + F_{i-1}^{j+1} - 2F_i^{j+1}}{\Delta x^2} + D_1 \frac{F_{i+1}^{j+1} - F_{i-1}^{j+1}}{2\Delta x} + F_i^{j+1} V_i^{j+1/2} + U_i^{j+1/2} \Bigg)\end{aligned} \tag{2.1.2}$$

Ricordo che l'equazione è calcolata all'istante $j + 1/2$ e che faccio le medie delle derivate spaziali e della funzione tra gli istanti j e $j + 1$, mentre le Funzioni esterne V e U le calcolo in $j + 1/2$, dato che sono note. Salto i passaggi e indico con $\eta = \frac{D_2 \Delta t}{\Delta x^2}$, posso scrivere:

$$\begin{aligned}\left(-\eta + D_1 \frac{\Delta t}{2\Delta x}\right) F_{i-1}^{j+1} + \left(2 + 2\eta - \Delta t V_i^{j+1/2}\right) F_i^{j+1} + \left(-\eta - D_1 \frac{\Delta t}{2\Delta x}\right) F_{i+1}^{j+1} - \Delta t U_i^{j+1/2} = \\ \left(\eta - D_1 \frac{\Delta t}{2\Delta x}\right) F_{i-1}^{j} + \left(2 - 2\eta + \Delta t V_i^{j+1/2}\right) F_i^{j} + \left(\eta + D_1 \frac{\Delta t}{2\Delta x}\right) F_{i+1}^{j} + \Delta t U_i^{j+1/2}\end{aligned} \tag{2.1.3}$$

Per rendere più chiara spiegazione e risoluzione della matrice tridiagonale riassumo l'equazione precedente in:

$$a_i^j F_{i-1}^{j+1} + d_i^j F_i^{j+1} + c_i^j F_{i+1}^{j+1} + e_i^j = ak_i^j F_{i-1}^j + dk_i^j F_i^j + ck_i^j F_{i+1}^j + ek_i^j \tag{2.1.4}$$

con:

$$\begin{array}{llll} a_i^j = -1 + \frac{D_1}{D_2}\frac{\Delta x}{2} & ak_i^j = 1 - \frac{D_1}{D_2}\frac{\Delta x}{2} & & \text{parametri dei } (F_{i-1}^*) \\ d_i^j = \frac{1}{\eta}\left(2 - \Delta t V_i^{j+1/2}\right) + 2 & dk_i^j = \frac{1}{\eta}\left(2 + \Delta t V_i^j + 1/2\right) - 2 & & \text{parametri dei } (F_i^*) \\ c_i^j = -1 - \frac{D_1}{D_2}\frac{\Delta x}{2} & ck_i^j = 1 + \frac{D_1}{D_2}\frac{\Delta x}{2} & & \text{parametri dei } (F_{i+1}^*) \\ e_i^j = -\frac{\Delta x^2}{D_2} U_i^{j+1/2} & ek_i^j = \frac{\Delta x^2}{D_2} U_i^{j+1/2} & & \text{funzioni esterne} \end{array} \tag{2.1.5}$$

2.2 La matrice Tridiagonale: soluzione

Per ogni istante di tempo j ho un sistema di N equazioni nella forma:

$$a_i F_{i-1}^{j+1} + d_i F_i^{j+1} + c_i F_{i+1}^{j+1} + e_i = ak_i F_{i-1}^j + dk_i F_i^j + ck_i F_{i+1}^j + ek_i \tag{2.2.1}$$

Dove j rappresenta l'istante di tempo che conosco e $j + 1$ quello che sto calcolando. La prima cosa da fare è portare nel membro a destra tutte le cose che conosco, dando per scontato che l'unica incognita dell'equazione è la funzione:

$$a_i F_{i-1}^{j+1} + d_i F_i^{j+1} + c_i F_{i+1}^{j+1} = ak_i F_{i-1}^j + dk_i F_i^j + ck_i F_{i+1}^j + ek_i - e_i \tag{2.2.2}$$

Per proseguire con la risoluzione è meglio passare alla rappresentazione matriciale del sistema (rappresento gli N punti rispettando le convenzioni del C, quindi $i = 0 \to N-1$):

$$\begin{pmatrix} d_0 & c_0 & & & & \\ a_1 & d_1 & c_1 & & & \\ & & ... & & & \\ & & & a_{N-2} & d_{N-2} & c_{N-2} \\ & & & & a_{N-1} & d_{N-1} \end{pmatrix} F^{j+1} = \begin{pmatrix} dk_0 & ck_0 & & & & \\ ak_1 & dk_1 & ck_1 & & & \\ & & ... & & & \\ & & & ak_{N-2} & dk_{N-2} & ck_{N-2} \\ & & & & ak_{N-1} & dk_{N-1} \end{pmatrix} F^j + \begin{pmatrix} ek_0 - e_0 \\ ek_1 - e_1 \\ ... \\ ek_{N-2} - e_{N-2} \\ ek_{N-1} - e_{N-1} \end{pmatrix} \tag{2.2.3}$$

Per comodità compatto il lato conosciuto in un vettore B^j le cui componenti sono:

$$b_i^j = ak_i F_{i-1}^j + dk_i F_i^j + ck_i F_{i+1}^j + ek_i - e_i \tag{2.2.4}$$

$$\begin{pmatrix} d_0 & c_0 & & & & \\ a_1 & d_1 & c_1 & & & \\ & & ... & & & \\ & & & ... & & \\ & & & a_{N-2} & d_{N-2} & c_{N-2} \\ & & & & a_{N-1} & d_{N-1} \end{pmatrix} F^{j+1} = B^j \tag{2.2.5}$$

A questo punto procedo con il trasformare la matrice nella somma di una matrice identità e di una matrice con valori non nulli solo nelle celle sopra alla diagonale, faccio vedere i primi passaggi:

$$\begin{pmatrix} d_0 & c_0 & 0 & ... & 0 \\ a_1 & d_1 & c_1 & ... & 0 \\ & . & . & . & \end{pmatrix} F^{j+1} = \begin{pmatrix} b_0 \\ b_1 \\ ... \end{pmatrix} \to \begin{pmatrix} 1 & \frac{c_0}{d_0} & 0 & ... & 0 \\ a_1 & d_1 & c_1 & ... & 0 \\ & . & . & . & \end{pmatrix} F^{j+1} = \begin{pmatrix} \frac{b_0}{d_0} \\ b_1 \\ ... \end{pmatrix} \tag{2.2.6}$$

Proseguendo, chiamando $h_0 = \frac{c_0}{d_0}$ e $p_0 = \frac{b_0}{d_0}$, moltiplico la prima riga per a_1 e la sottraggo alla seconda, in modo da eliminare a_1 dalla seconda riga:

$$\begin{pmatrix} 1 & h_0 & 0 & ... & 0 \\ a_1 - a_1 & d_1 - a_1 h_0 & c_1 & ... & 0 \\ & . & . & . & \end{pmatrix} F^{j+1} = \begin{pmatrix} p_0 \\ b_1 - a_1 p_0 \\ ... \end{pmatrix} \to \begin{pmatrix} 1 & h_0 & 0 & ... & 0 \\ 0 & 1 & \frac{c_1}{d_1 - a_1 h_0} & ... & 0 \\ & . & . & . & \end{pmatrix} F^{j+1} = \begin{pmatrix} p_0 \\ \frac{b_1 - a_1 p_0}{d_1 - a_1 h_0} \\ ... \end{pmatrix} \tag{2.2.7}$$

A questo punto chiamo $h_1 = \frac{c_1}{d_1 - a_1 h_0}$ e $p_1 = \frac{b_1 - a_1 p_0}{d_1 - a_1 h_0}$ e ripeto il ragionamento precedente sottraendo la seconda riga alla terza:

$$\begin{pmatrix} 1 & h_0 & 0 & ... & 0 \\ 0 & 1 & h_1 & ... & 0 \\ 0 & a_3 - a_3 & d_3 - a_3 h_1 & c_3 & ... \\ & . & . & . & \end{pmatrix} F^{j+1} = \begin{pmatrix} p_0 \\ p_1 \\ b_3 - a_3 p_1 \\ ... \end{pmatrix} \to \begin{pmatrix} 1 & h_0 & 0 & ... & 0 \\ 0 & 1 & h_1 & ... & 0 \\ 0 & 0 & 1 & \frac{c_3}{d_3 - a_3 h_1} & ... \\ & . & . & . & \end{pmatrix} F^{j+1} = \begin{pmatrix} p_0 \\ p_1 \\ \frac{b_3 - a_3 p_1}{d_3 - a_3 h_1} \\ ... \end{pmatrix} \tag{2.2.8}$$

A questo punto chiamo $h_3 = \frac{c_3}{d_3 - a_3 h_1}$ e $p_3 = \frac{b_3 - a_3 p_1}{d_3 - a_3 h_1}$ e proseguo, ottengo così le regole:

$$h_i = \frac{c_i}{d_i - a_i h_{i-1}} \tag{2.2.9}$$

e

$$p_i = \frac{b_i - a_i p_{i-1}}{d_i - a_i h_{i-1}} \tag{2.2.10}$$

A questo punto ho semplificato il sistema:

(2.2.11)

$$\begin{pmatrix} 1 & h_0 & & & \\ & 1 & h_1 & & \\ & & ... & & \\ & & & ... & \\ & & & 1 & h_{N-2} \\ & & & & 1 \end{pmatrix} F^{j+1} = P$$

(2.2.12)

Per risolvere il sistema devo calcolare il vettore delle P per poi ottenere i valori di F^{j+1} a partire dall'ultimo $F_{N-1}^{j+1} = p_{N-1}$ con la formula:

$$F_i^{j+1} = p_i + h_i F_{i+1}^{j+1} \tag{2.2.13}$$

A questo punto ho bisogno di conoscere come trattare le condizioni al contorno.

3 Condizioni al contorno

In seguito espongo come è possibile adottare alcune condizioni al contorno:

- Dirichlet: Conosco i valori della funzione negli estremi del dominio
- Neumann: Conosco i valori della derivata della funzione negli estremi del dominio
- Robin: Conosco una combinazione lineare tra il valore della funzione e la sua derivata negli estremi del dominio
- Miste: Negli estremi ho tipi differenti di condizioni al contorno

3.1 Dirichlet

Conosco il valore della funzione negli estremi del dominio.

$$F(x,t) = f(x,t) \forall x \in \partial D \tag{3.1.1}$$

Assegno a F_0^{j+1} e F_{N-1}^{j+1} il valore noto, e' quindi inutile calcolare la prima e l'ultima riga della matrice $N \times N$ e posso trattare tutto come se la matrice fosse $N-2 \times N-2$, con indici da 1 a $N-2$ per tenere conto delle condizioni devo cambiare i valori:

$$\begin{array}{llll|llll} a'_0 = 0 & ak'_0 = 0 & a'_1 = 0 & ak'_1 = ak_1 \\ d'_0 = 1 & dk'_0 = 0 & d'_1 = d_1 & dk'_1 = dk_1 \\ c'_0 = 0 & ck'_0 = 0 & c'_1 = c_1 & ck'_1 = ck_1 \\ e'_0 = -F_0^{j+1} & ek'_0 = 0 & e'_1 = e_1 + a_1 F_0^{j+1} & ek'_1 = ek_1 \end{array} \tag{3.1.2}$$

Che equivale a scrivere:

$$\begin{array}{lll} b'_0 = F_0^{j+1} & h'_0 = 0 & p'_0 = F_0^{j+1} \\ b'_1 = b_1 - a_1 F_0^{j+1} & h'_1 = \frac{c_1}{d'_1} & p'_1 = \frac{b'_1}{d'_1} \end{array} \tag{3.1.3}$$

Di conseguenza $F_1^{j+1} = p'_1 + h'_1 F_2^{j+1}$ e $F_0^{j+1} = p'_0 + h'_0 F_1^{j+1} = F_0^{j+1}$. Mentre se la condizione si presenta per l'ultimo punto del dominio:

$$\begin{array}{llll|llll} a'N-2 = a_{N-2} & ak'_{N-2} = ak_{N-2} & a'_{N-1} = 0 & ak'_{N-1} = 0 \\ d'_{N-2} = d_{N-2} & dk'_{N-2} = dk_{N-2} & d'_{N-1} = 1 & dk'_{N-1} = 0 \\ c'_{N-2} = 0 & ck'_{N-2} = ck'_{N-2} & c'_{N-1} = 0 & ck'_{N-1} = 0 \\ e'_{N-2} = e_{N-2} + c_{N-2} F_{N-1}^{j+1} & ek'_{N-2} = -F_{N-1}^{j+1} & e'_{N-1} = -F_{N-1}^{j+1} & ek'_{N-1} = 0 \end{array} \tag{3.1.4}$$

Che posso riscrivere:

$$\begin{array}{lll} b'_{N-2} = b_{N-2} - c_{N-2} F_{N-1}^{j+1} & h'_{N-2} = 0 & p'_{N-2} = p_{N-2} \\ b'_{N-1} = F_{N-1}^{j+1} & h'_{N-1} = 0 & p'_{N-1} = F_0^{j+1} \end{array} \tag{3.1.5}$$

e di conseguenza $F_{N-1}^{j+1} = p'_{N-1} = F_{N-1}^{j+1}$ e $F_{N-2}^{j+1} = p'_{N-2} + h'_{N-2} F_{N-1}^{j+1} = p'_{N-2}$.

3.2 Neumann

Conosco il valore della derivata negli estremi del dominio.

$$\frac{\partial F}{\partial x}(x,t) = f(x,t) \forall x \in \partial D \tag{3.2.1}$$

La spiegazione e l'esempio per questa risoluzione lo fornisco nel paragrafo dedicato a Robin.

3.3 Robin

Conosco una combinazione lineare tra la derivata e il valore della funzione negli estremi del dominio.

$$\frac{\partial F}{\partial x}(x,t) + r(x,t) F(x,t) = f(x,t) \forall x \in \partial D \tag{3.3.1}$$

Come prima, per mantenere la precisione del metodo (Δx^2) non posso usare la definizione centrale, ho quindi bisogno di inventarmi un "nodo fantasma" F_{-1}^{j+1} (o F_N^{j+1} se fosse la condizione nell'ultimo punto del dominio):

$$\frac{\partial F}{\partial x}(x(i=0), t(j=j+1)) = \frac{F_1^{j+1} - F_{-1}^{j+1}}{2\Delta x} \tag{3.3.2}$$

Prendo la (3.3.1) e faccio le adeguate sostituzioni per $i = 0$ e al tempo geerico $j = n$:

$$\frac{F_1^n - F_{-1}^n}{2\Delta x} + R_0^n F_0^n = f_0^n \rightarrow F_{-1}^n = F_1^n + 2\Delta x \left(R_0^n F_0^n - f_0^n\right) \tag{3.3.3}$$

Partendo dalla forma matriciale del problema generico:

$$a_0 F_{-1}^{j+1} + d_0 F_0^{j+1} + c_0 F_1^{j+1} + e_0 = ak_0 F_{-1}^j + dk_0 F_0^j + ck_0 F_1^j + ek_0 \tag{3.3.4}$$

e sostituendo i vari F_{-1}^n:

$$a_0 \left(F_1^{j+1} + 2\Delta x \left(R_0^{j+1} F_0^{j+1} - f_0^{j+1}\right)\right) + d_0 F_0^{j+1} + c_0 F_1^{j+1} = ak_0 \left(F_1^j + 2\Delta x \left(R_0^j F_0^j - f_0^j\right)\right) + dk_0 F_0^j + ck_0 F_1^j \tag{3.3.5}$$

A questo punto raccolgo i termini con lo stesso punto della funzione:

$$\left(d_0 + 2a_0 R_0^{j+1} \Delta x\right) F_0^{j+1} + (c_0 + a_0) F_1^{j+1} - 2a_0 f_0^{j+1} \Delta x = \left(dk_0 + 2ak_0 R_0^j \Delta x\right) F_0^j + (ck_0 + ak_0) F_1^j - 2ak_0 f_0^j \Delta x \tag{3.3.6}$$

E quindi i parametri interessati della matrice diventano:

$$\begin{array}{ll} a_0' = 0 & ak_0' = 0 \\ d_0' = d_0 + 2a_0 R_0^{j+1} \Delta x & dk_0' = dk_0 + 2ak_0 R_0^j \Delta x \\ c_0' = a_0 + c_0 & ck_0' = ak_0 + ck_0 \\ e_0' = e_0 - 2a_0 f_0^{j+1} \Delta x & ek_0' = ek_0 - 2ak_0 f_0^j \Delta x \end{array} \tag{3.3.7}$$

Che equivale a scrivere:

$$\begin{array}{l} b_0' = dk_0' F_0^j + ck_0' F_1^j + 2\Delta x \left(a_0 f_0^{j+1} - ak_0 f_0^j\right) \\ h_0' = \frac{c_0'}{d_0'} \\ p_0' = \frac{b_0'}{d_0'} \end{array} \tag{3.3.8}$$

e di conseguenza $F_0^{j+1} = p_0' + h_0' F_1^{j+1}$.

Mentre se la condizione si presenta come ultimo punto del dominio:

$$\frac{F_N^n - F_{N-2}^n}{2\Delta x} + R_{N-1}^n F_{N-1}^n = f_{N-1}^n \rightarrow F_N^n = F_{N-2}^n - 2\Delta x \left(R_{N-1}^n F_{N-1}^n - f_{N-1}^n\right) \tag{3.3.9}$$

Salto i passaggi, molto simili a quelli della spiegazione precedente, per il calcolo di b_{N-1}' andranno usati i seguenti parametri:

$$\begin{array}{ll} a_{N-1}' = a_{N-1} + c_{N-1} & ak_{N-1}' = ak_{N-1} + ck_{N-1} \\ d_{N-1}' = d_{N-1} - 2c_{N-1} R_{N-1}^{j+1} \Delta x & dk_{N-1}' = dk_{N-1} - 2ck_{N-1} R_{N-1}^j \Delta x \\ c_{N-1}' = 0 & ck_{N-1}' = 0 \\ e_{N-1}' = e_{N-1} + c_{N-1} f_{N-1}^{j+1} \Delta x & ek_{N-1}' = ek_{N-1} + ck_{N-1} f_{N-1}^{j+1} \Delta x \end{array} \tag{3.3.10}$$

Che equivale a scrivere:

$$\begin{array}{l} b_{N-1} = ak_{N-1}' F_{N-2}^j + dk_{N-1}' F_{N-1}^j - 2\Delta x \left(c_{N-1} f_{N-1}^{j+1} - ck_{N-1} f_{N-1}^j\right) \\ h_{N-1}' = 0 \\ p_{N-1}' = \frac{b_{N-1}' + a_{N-1}' p_{N-2}}{d_{N-1}' - a_{N-1}' h_{N-2}} \end{array} \tag{3.3.11}$$

e di conseguenza $F_{N-1}^{j+1} = p_{N-1}'$, anche perchè è il primo punto da cui si parte per calcolare il valore della funzione in $j + 1$.

Se faccio in modo di eliminare il coefficiente che moltiplica il valore della funzione (gli R) ottengo le condizioni a contorno di Neuman.

3.4 Osservazione

Il modo in cui ho trattato i parametri per quanto riguarda le condizioni al contorno di Dirichlet nei punti 0 e $N-1$, non è matematicamente correttissimo infatti i parametri andrebbero messi tutti a 0 in quanto quei punti non fanno parte dell'algoritmo. Ho impostato i valori per avere un algoritmo che possa svolgere il calcolo rispettando le condizioni al contorno senza sapere quali siano, mettendo nelle mani dell'utente che si occuperà di impostare i corretti parametri della matrice la gestione delle condizioni.

4 Applicazione

4.1 Equazione del calore

Riprendiamo l'equazione del calore:

$$\frac{\partial T}{\partial t}(x,t) = k\frac{\partial^2}{\partial x^2}T(x,t) \tag{4.1.1}$$

Per rispettare la convenzione di prima $D_2 = k$, $D_1 = U = V(x,t) = 0$.
A questo punto posso sostituire, con $\eta = k\frac{\Delta t}{\Delta x^2}$:

$$\begin{array}{ll} a_i^j = -1 & ak_i^j = 1 \\ d_i^j = \frac{2}{\eta} + 2 & dk_i^j = \frac{2}{\eta} - 2 \\ c_i^j = -1 & ck_i^j = 1 \\ e_i^j = 0 & ek_i^j = 0 \end{array} \tag{4.1.2}$$

e procedere con i calcoli.

4.2 Equazione di Schrödinger

Lavoriamo con l'equazione di Schrödinger dipendente dal tempo:

$$i\hbar\frac{\partial \psi}{\partial t}(x,t) = \left[-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x,t)\right]\psi(x,t) \tag{4.2.1}$$

prima di tutto portiamola in una forma tale che non ci sia nulla a moltiplicare la derivata temporale:

$$\frac{\partial \psi}{\partial t}(x,t) = \left[i\frac{\hbar}{2m}\frac{\partial^2}{\partial x^2} + \frac{\Lambda(x,t)}{i\hbar}\right]\psi(x,t) \tag{4.2.2}$$

Per rispettare la convenzione di prima $D_2 = i\frac{\hbar}{2m}$, $D_1 = U = 0$ e $V(x,t) = \frac{\Lambda(x,t)}{i\hbar}$. Ho usato Λ per indicare il potenziale in modo da evitare confusione.
A questo punto posso sostituire, con $\eta = i\frac{\hbar}{2m}\frac{\Delta t}{\Delta x^2}$:

$$\begin{array}{ll} a_i^j = -1 & ak_i^j = 1 \\ d_i^j = \frac{1}{\eta}\left(2 - \Delta t V_i^{j+1}\right) + 2 & dk_i^j = \frac{1}{\eta}\left(2 + \Delta t V_i^j\right) - 2 \\ c_i^j = -1 & ck_i^j = 1 \\ e_i^j = 0 & ek_i^j = 0 \end{array} \tag{4.2.3}$$

5 Il programma

Per semplicità e alleggerire i calcoli ho impostato $\hbar = 1$, Gli eseguibili compilabili dal Makefile sono i seguenti, nei commenti è spiegato come eseguirli:

```
#https://root.cern.ch/interacting-shared-libraries-rootcint
#risolve un'equazione di Schodinger: come primo argonento il file del potenziale
#come secondo, opzionale, il file che indica le CI, di base e' gauss.set
single: MainC.cpp TridiagC.o CrankSolverC.o impostazioniC.o experimentC.o
	@echo Compilo $@
	@$(CC11) $(CFLAGS) -o $@ $^  -DUSECOMPLEX
#disegna soluzione, errore e pesi dell'integrale del file di dati in argomento
drawer:drawer.cpp preparedraw.o
	@echo Compilo $@
	@$(CC11) $(CFLAGS) -o $@ $^ $(LIBROOT) $(CFLAGSROOT)
#controlla i .dat letti in namelist.txt
analisi:analisi.cpp preparedraw1D.o
	@echo Compilo $@
	@$(CC11) $(CFLAGS) -o $@ $^ $(LIBROOT) $(CFLAGSROOT)
#si comporta come maincrankC, solo che carica i nomi dei file del potenziale da
#namelist.txt, e come argomento opzionale ha il file delle CI
experiment: Experiments.cpp TridiagC.o CrankSolverC.o impostazioniC.o experimentC.o
	@echo Compilo $@
	@$(CC11) $(CFLAGS) -o $@ $^  -DUSECOMPLEX
#visualizza vari potenziali in diversi files.
CVD: CVDrawer.cpp impostazioniC.o TridiagC.o
	@echo Compilo $@
	@$(CC11) $(CFLAGS) -o $@ $^  -DUSECOMPLEX $(LIBROOT) $(CFLAGSROOT)
#una interfaccia grafica per impostare un esperimento alla volta
preview: preview.cpp visual.o libVisual.so
	@echo Compilo $@
	@$(CC11) $(CFLAGS) -o $@ -DSTANDALONE $^ $(LIBROOT) $(CFLAGSROOT)
```

Il programma più semplice ($> make single) è composto da un blocco *impostazioni* che carica i file con le informazioni sulla simulazione, crea la trimatrice e le condizioni iniziali ,e un blocco *CrankSolver* che si occupa di svolgere risolvere il sistema passo per passo (nel nostro caso non avendo potenziali che dipendono dal tempo la riduzione della trimatrice viene fatta solo una volta, nella costruzione del *CrankSolver*). Ho scelto di impostare il *CrankSolver* in modo che contenga solo i punti del passo attuale perché l'elevato numero di punti rischia di intasare la memoria del sistema piuttosto in fretta: i punti calcolati vengono salvati su un file in una cartella "./results" che deve essere creata prima di lanciare l'eseguibile, il numero di punti e la frequenza di essi viene deciso nei file di impostazione.

5.1 Impostazioni

Il programma più semplice, compilabile con $> make single carica i dati da tre file di impostazioni, in cui viene alternata una riga di commento, anticipata da un "#" e una con i valori dell'impostazione, l'ordine è importante e il commento non deve contenere spazi.

Il file generale delle impostazioni:

Listing 1: Impostazioni principali

```
#massa
1
#lunghezza
60
#tmax
10
#Ndipende_da_intervallo=0
0
```

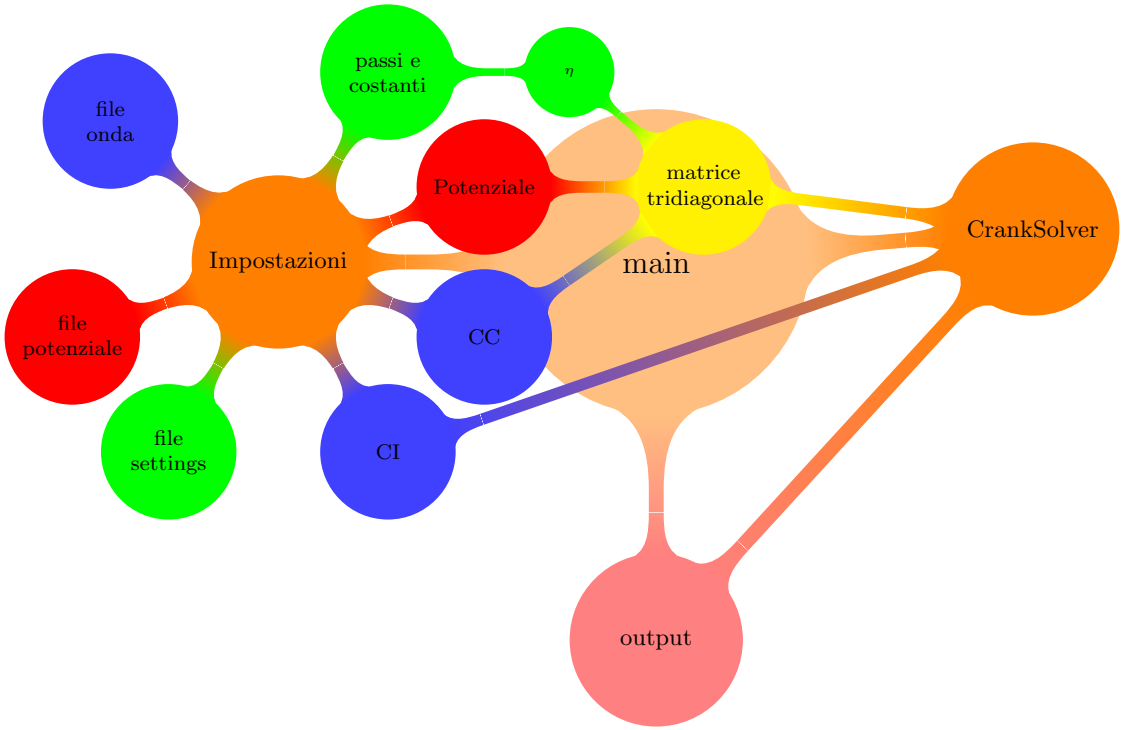


Figura 1: Collegamenti tra i blocchi del programma

```
#Nl
10000
#spacestep.
0.0015
#Nt
1000
#timestep
0.001
#timeskip
100
#spaceskip
1
#precisione
5e-3
```

In questo file fornisco i parametri principali della simulazione, quelli che ho previsto che avrei cambiato più di rado durante la simulazione. La massa, la lunghezza sono ovvi, tmax è il tempo a cui si ferma la simulazione. L'opzione "*#Ndipende_da_intervallo=0*" se impostata uguale a 0 fa si' che il numero di passi (spaziali e temporali) venga calcolato conoscendo la lunghezza totale e il passo, se impostata diversa da 0 invece fa sì che il programma calcoli il passo a partire dalla lunghezza totale e il numero di punti voluti. Le opzioni "*#timeskip*" e "*#spaceskip*" servono nel momento di salvare i risultati: il primo imposta ogni quanti passi temporali esportare il vettore di punti, che vengono salvati ad intervalli decisi dal secondo parametro. Il parametro "*#precisione*" invece blocca la simulazione prima del tempo massimo se l'errore (vedi sottosezione 5.2) supera il valore impostato, se lo si imposta a 0 salta il controllo.

Il file delle condizioni iniziali:

Listing 2: Impostazioni CI

```
#Bump_Gauss
G
#impostazioni_onda_norm_stdev_midpoint
1 2 5
#impostazionicc0_012DNR_val_pesorobin
```

```
2 0 0
#impostazioniccN_012DNR_val_pesorobin
2 0 0
#Energia
10
```

Il primo parametro deve essere una lettera e serve a indicare al programma la forma del pacchetto d'onda nelle condizioni iniziali, se viene scritto "b" il programma disegnerà la funzione "bump" con parametri presi dalla riga successiva, nell'ordine "altezza del massimo", "larghezza" e "punto del massimo"; negli altri casi una gaussiana con i parametri "altezza", "deviazione standard" e "punto medio". I due set successivi alle impostazioni dell'onda riguardano le condizioni al contorno, la prima che può essere 0, 1 o 2 indica il tipo di condizione, rispettivamente Dirichlet, Neumann o Robin sulla stessa riga il primo numero indica il valore della funzione, della derivata o della combinazione lineare, il terzo parametro viene solo utilizzata da Robin ed è il peso della funzione. L'ultima impostazione è l'energia che si vuole impostare per l'onda.

Il file del potenziale:

Listing 3: Impostazioni potenziale

```
#tipo_Gauss_Muro_Salto
M
#impostazioni_Val_Pos_Vpar
10 30 4
```

Il primo parametro imposta la forma del potenziale, che può essere una gaussiana, un "bump", un rettagolo oppure un salto. Nella riga successiva il primo parametro imposta l'altezza del punto più alto del potenziale, se messo a 0 dice all'algoritmo di calcolare l'evoluzione in assenza di potenziale, il secondo parametro indica il centro della funzione se è pari, oppure la posizione del salto, l'ultimo parametro indica la larghezza o la deviazione standard del potenziale non viene utilizzato se la forma è il salto.

5.2 Lancio senza potenziale: errore

Prima di tutto voglio effettuare qualche simulazione senza potenziale per poter osservare come si comporta l'onda.

Ho bisogno di definire un "errore" per capire quanto la simulazione possa essere andata a buon fine: la mia scelta è stata di utilizzare la norma della funzione, ricordando che le funzioni d'onda sono L^2 e quindi la loro norma è $|f|^2 = \int_X |f|^2 \, d\mu$, utilizzando la definizione dell'integrale discreto di Simpson, il che mi limita a poter fare solo simulazioni con un numero di passi pari (e in particolare multipli di 4 per aver un integrale preciso anche quando calcolo l'integrale solo nella prima o seconda metà del dominio) nello spazio.

Nella simulazione blocco il calcolo se la norma della funzione si discosta troppo dalla norma delle condizioni iniziali.

Per capire come impostare le simulazioni ho effettuato diversi lanci con varie combinazioni di passo temporale e passo spaziale. Riporto i vari grafici degli errori. Da questi lanci capisco che devo scartare l'idea di usare un passo temporale grande in quanto ho notato che le funzioni d'onda vengono "rallentate", come si può vedere in Figura 2. Dalle figure noto che le simulazioni devono avere come massimo passo temporale 0.01. Il punto in cui gli andamenti perdono l'andamento lineare e' dovuto al fatto che il pacchetto sta impattando contro il bordo del dominio e quindi inizia a rimbalzare.

Dato che ho simulato onde con una energia di $10udm$ mi aspetto che la velocità sia:.....

Poi ho quindi guardato l'errore di alcune simulazioni. Scarto a priori i passi più grandi di 0.01.

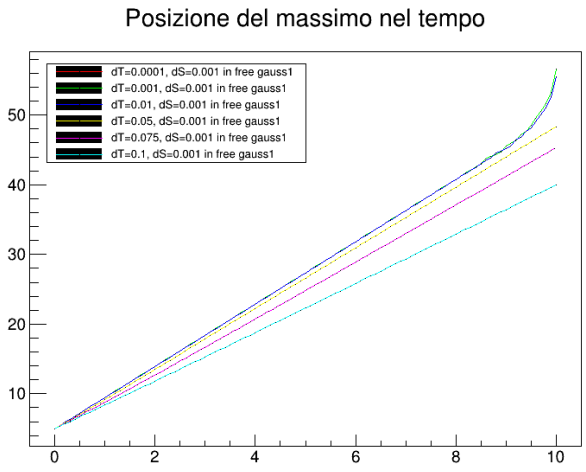
A questo punto ho fatto una simulazione diminuendo il tempo a 6 in modo da non avere a che fare con l'esplosione dell'errore dovuta al limite del dominio. e cosi' da poter osservare quale fosse il miglior passo spaziale. Ho fatto i lanci con due diverse condizioni al contorno (gaussiano con σ 1 e 2).

Osservando Figura 4 utilizzare come passi spaziali e temporali valori compresi tra 0.01 e 0.001 ha un buon rapporto errore(il cui modulo è inferiore a 10^-7)/peso sul calcolatore, in particolare si nota che il passo spaziale pesa di più sull'errore. Ho scelto di continuare con $dT = 0.01$ e $dS = 0.005$ in quanto hanno errore molto simile a $dS = 0.001$ (Pag. 15) ma sono più rapidi nei calcoli (sulla mia macchina).

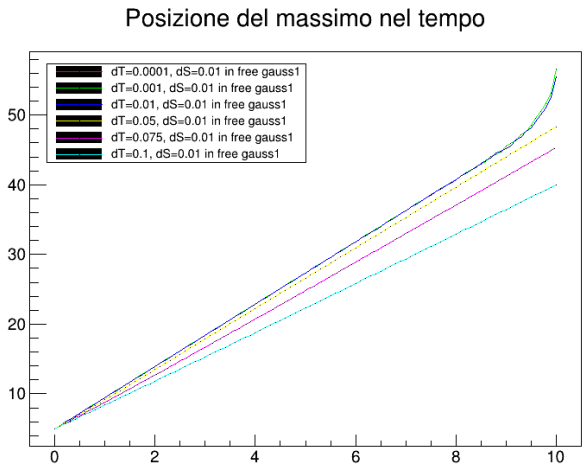
5.3 Condizioni al contorno

Quello che io voglio simulare è un pacchetto d'onda che sia partito da $-\infty$ e arrivi nel "punto interessante", che è l'area che osservo nella simulazione e poi prosegua fino all'infinito (o venga eventualmente riflesso).

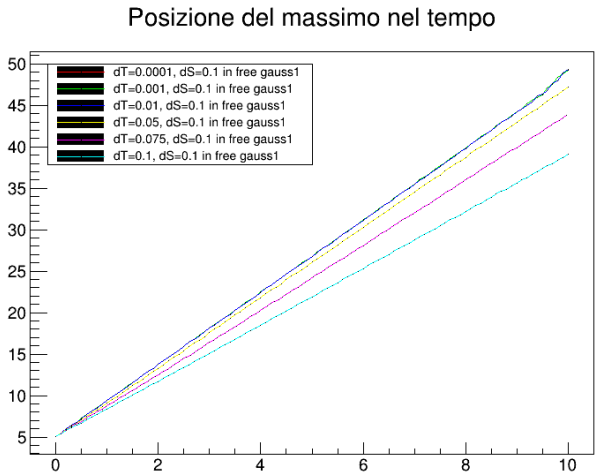
Per quanto riguarda le condizioni al contorno ho avuto idee:



(a) Le differenze per $dS = 0.001$



(b) Le differenze per $dS = 0.01$



(c) Le differenze per $dS = 0.1$

Figura 2: Le posizioni dei massimi nel tempo della simulazione a vari passi temporali e spaziali

- Non tengo conto delle condizioni al contorno, faccio in modo di usare pacchetti d'onda ristretti e blocco la simulazione quando questa va ad avvicinarsi troppo ai bordi.
- Uso una forma d'onda tale per cui conosco la funzione e la sua derivata in ogni punto
- Emulo un ambiente chiuso (con ai lati muri di potenziale alti ∞)

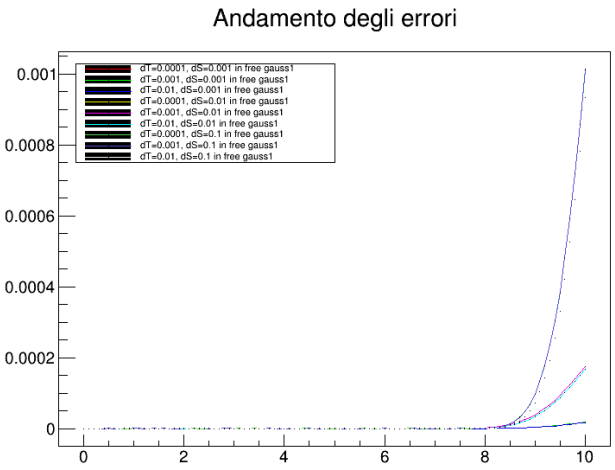
La prima ipotesi è stata quella che ho usato in tutte le prove che ho effettuato prima di iniziare a raccogliere dati.

Per la seconda ipotesi dovrei usare come condizione iniziale una funzione e calcolarne la derivata per ogni passo negli estremi a priori il discorso non si applica ma i paccheti d'onda hanno una dispersione tendono ad allargarsi, per cui se per esempio avessi un pacchetto d'onda generico ($\Psi(x,t) = e^{ikx}\psi(x,t)$)la relazione dovrebbe essere $\partial_x\Psi(x,t) = ik\Psi(x,t) + e^{ikx}\partial_x\psi(x,t)$ e potrei quindi utilizzare le CC di Robin in caso conoscessi come la funzione si disperde nel tempo potrei ricalcolare ad ogni passo le condizioni al contorno ed applicarle. Questo però mi porterebbe a svariati problemi quali l'incertezza che la simulazione e la teoria siano esattamente parallele sulla dispersione

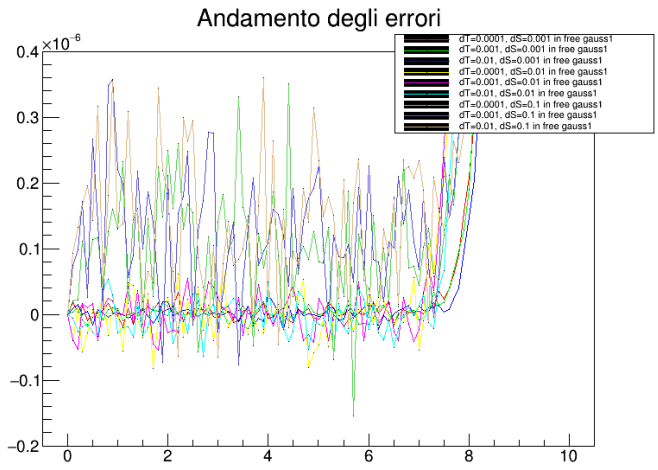
La terza idea equivale a utilizzare Dirichlet con il valore della funzione pari a 0 nei bordi. In pratica assomiglia alla mia prima idea, specie se blocco la simulazione prima dell'impatto

5.4 Dispersione e forme dei pacchetti d'onda

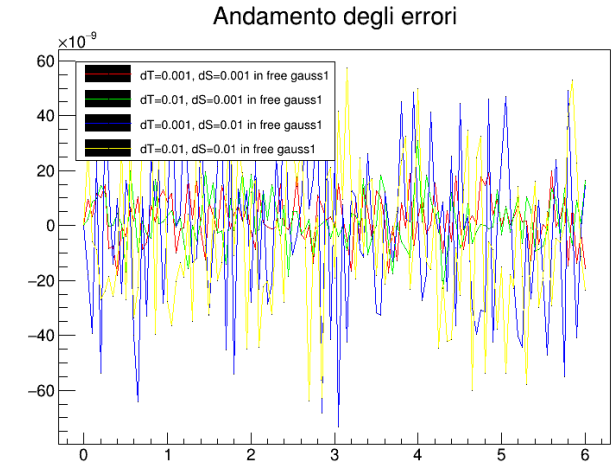
La prima cosa che si può osservare nelle simulazioni (e in particolare in Figura 6) è la dispersione dell'onda, ovvero il pacchetto si allarga nel tempo. Questa è una caratteristica dell'equazione di Schrödinger . Per ottenere la soluzione



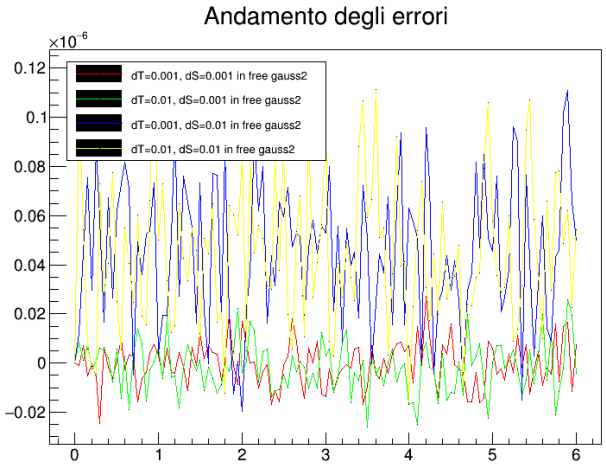
(a) Il grafico completo degli errori, si vede anche quanto la riflessione influsisca sul modulo della funzione



(b) Il grafico completo degli errori uno zoom sugli andamenti non influenzati dalla riflessione



(a) CI: gaussiana($\sigma = 2$)



(b) CI: gaussiana($\sigma = 2$)

Figura 4: I passi che mi se,bravano più opportuni a confronto

dell'equazione con condizioni iniziali $\psi(x,0) = e^{-\frac{x^2}{2\sigma^2}}$ si passa dalla trasformata di Fourier spaziale $\hat{\psi}(x,0) = \sqrt{2\pi\sigma^2}e^{-\frac{k^2\sigma^2}{2}}$, che porta ad avere la funzione, con dipendenza temporale:

$$\hat{\psi}(k,t) = \sqrt{2\pi\sigma^2}e^{-\frac{k^2\sigma^2}{2}}e^{-i\frac{E}{\hbar}t} = \sqrt{2\pi\sigma^2}e^{-\frac{k^2\sigma^2}{2}}e^{-i\frac{\hbar^2k^2/2m}{\hbar}t} = \sqrt{2\pi\sigma^2}e^{-k^2\frac{\left(\sigma^2+i\hbar t/m\right)}{2}} \tag{5.4.1}$$

e poi ritrasformando:

$$\psi(x,t) = \frac{\sigma}{\sqrt{\sigma^2 + i\hbar t/m}}e^{-\frac{k^2}{2(\sigma+i\hbar t/m)}} \tag{5.4.2}$$

quelle che rappresento sono i valori assoluti della funzione:

$$|\psi(x,t)| = \frac{\sigma}{\sqrt[4]{\sigma^4 + (\hbar t/m)^2}}e^{-\frac{k^2}{2}\frac{\sigma^2}{\sigma^4+(\hbar t/m)^2}} = \sqrt{\frac{\sigma_0}{\sigma(t)}}e^{-\frac{k^2}{2\sigma^2(t)}} \tag{5.4.3}$$

E, chiamando σ_0 il valore assegnato nelle condizioni iniziali, mi aspetto dalla simulazione che σ evolva come

$$\sigma(t) = \sqrt{\frac{\sigma_0^4 + \left(\frac{\hbar}{m}t\right)^2}{\sigma_0^2}} \tag{5.4.4}$$

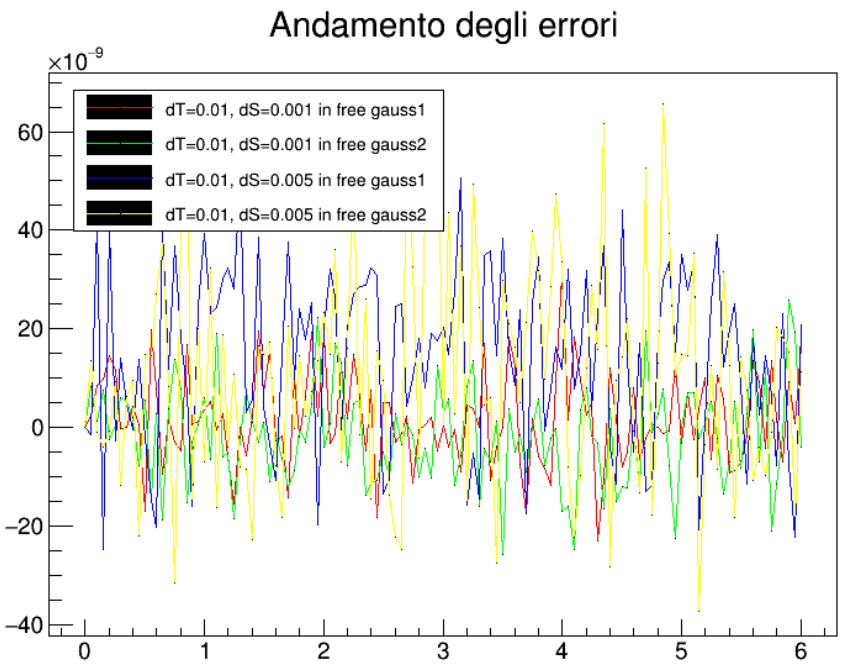


Figura 5: Gli errori dei passi che ritengo più convenienti

Ho fatto diversi fit, sul parametro σ, con diverse masse e diversi valori di σ iniziali, utilizzando come funzione da fittare (5.4.4). E l'algoritmo sembra funzionare per quanto riguarda la dispersione, come si può vede re in Figura 7, in Figura 8 e Figura 9 in dove la funzione ha energia diversa da 0

5.5 Condizioni iniziali

A questo punto provo anche a utilizzare la funzione "bump" come condizioni iniziale:

$$bump\,(x) = \begin{cases} e^{-\frac{1}{1-x^2}} & \text{per } |x| < 1 \\ 0 & \text{altrove} \end{cases} \tag{5.5.1}$$

In particolare io ho usato:

$$bump = (x) = \begin{cases} e^{-\frac{b^2}{b^2-(x-a)^2}} & \text{per } |x-a| < b \\ 0 & \text{altrove} \end{cases} \tag{5.5.2}$$

simulazione

5.6 Dispersione

teoria

simulazione

5.7 Potenziali

teoria

simulazione

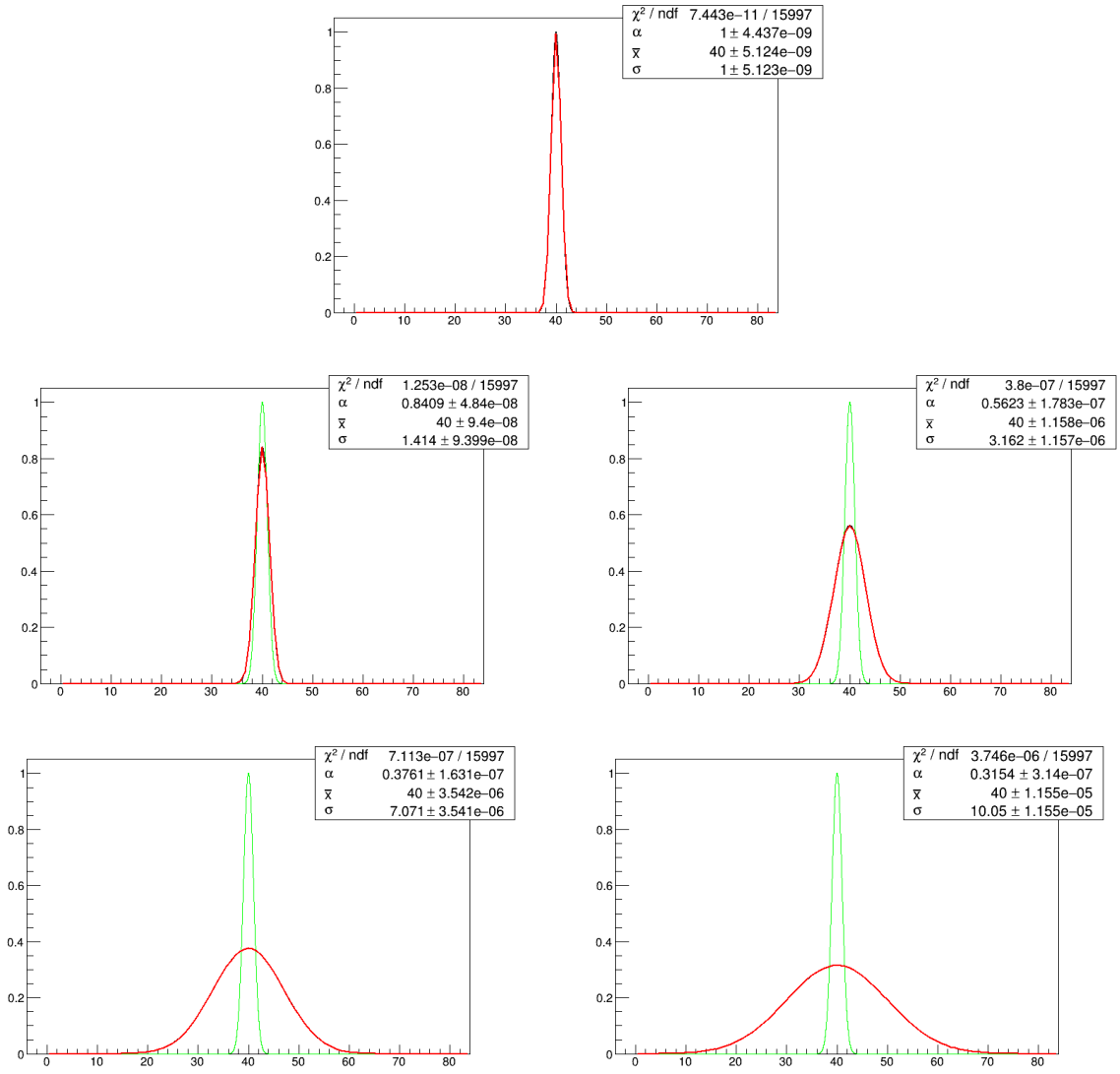
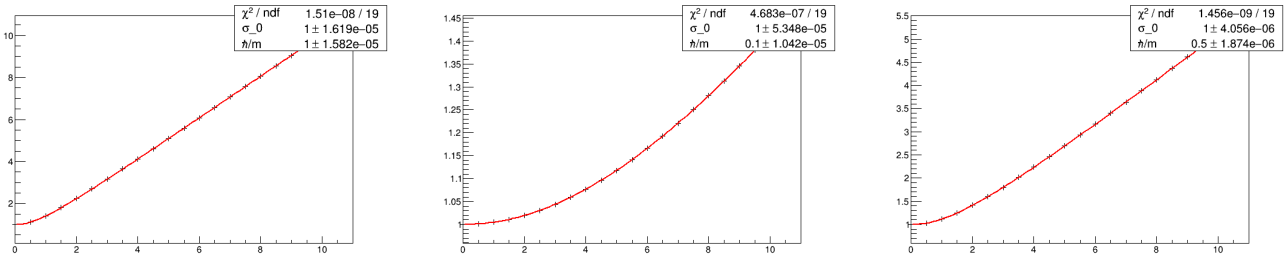
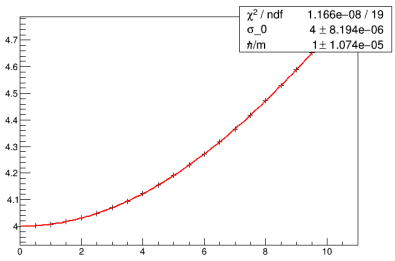


Figura 6: La dispersione delle gaussiana a $\sigma = 1$

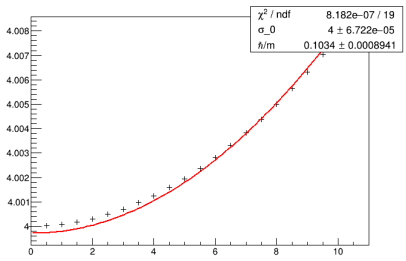


(a) $\sigma_0 = 1$, $m = 1$ (b) $\sigma_0 = 1$, $m = 10$ (c) $\sigma_0 = 1$, $m = 2$

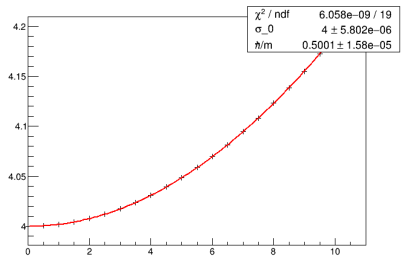
Figura 7: Alcuni fit dell'andamento della dispersione



(a) $\sigma_0 = 4$, $m = 1$

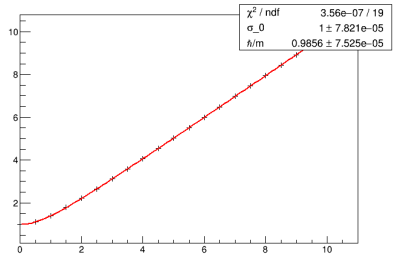


(b) $\sigma_0 = 4$, $m = 10$

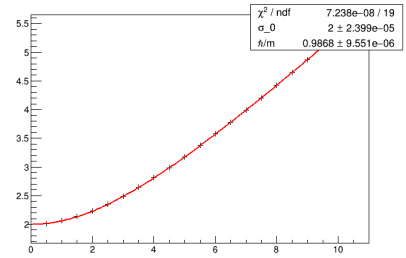


(c) $\sigma_0 = 4$, $m = 10$

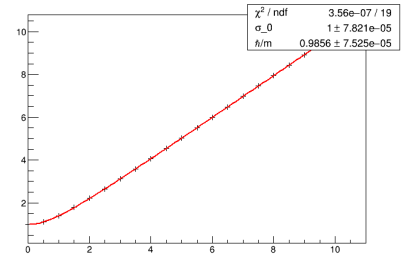
Figura 8: Alcuni fit dell'andamento della dispersione



(a) $\sigma_0 = 1$, $m = 1$, con $E = 10$



(b) $\sigma_0 = 2$, $m = 1$, con $E = 10$



(c) $\sigma_0 = 4$, $m = 1$, con $E = 10$

Figura 9: Alcuni fit dell'andamento della dispersione

t	α	$\bar{x}$	σ	t	α	$\bar{x}$	σ
0	1	40	1	5.5	0.422955	40	5.59001
0.5	0.94574	40	1.11803	6	0.405467	40	6.08259
1	0.840898	40	1.4142	6.5	0.389951	40	6.57628
1.5	0.744796	40	1.80272	7	0.376066	40	7.07086
2	0.668766	40	2.23596	7.5	0.363549	40	7.56615
2.5	0.609451	40	2.69242	8	0.352191	40	8.06202
3	0.562349	40	3.16219	8.5	0.341825	40	8.55837
3.5	0.524146	40	3.63995	9	0.332317	40	9.05512
4	0.492486	40	4.12299	9.5	0.323555	40	9.55221
4.5	0.465765	40	4.60964	10	0.315447	40	10.0496
5	0.442856	40	5.09887				

Tabella 1: I dati di una simulazione, con $\alpha = 1$, $\sigma = 1$ e $m = 1$

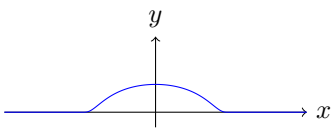


Figura 10: La funzione bump