Sistemi lineari

- Scrivere una funzione di Matlab che implementi il metodo iterativo del gradiente per l'approssimazione della soluzione di un sistema lineare $A\mathbf{x} = \mathbf{b}$:

$$
\begin{aligned}
&\mathbf{x}^0 \text{ assegnato} \\
&\text{per } k \geq 0 \\
&\quad \mathbf{r}^k = \mathbf{b} - A\mathbf{x}^k \\
&\quad \alpha_k = \frac{\mathbf{r}^{k\,T}\mathbf{r}^k}{\mathbf{r}^{k\,T}A\mathbf{r}^k} \\
&\quad \mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{r}^k
\end{aligned}
$$

Usare il test d'arresto basato sul residuo:

$$\text{se } \|\mathbf{r^k}\| < \texttt{toll}\ \|\mathbf{b}\| \text{ STOP.}$$

Sistemi lineari

- Per verificare che il programma funzione correttamenta risolvere il sistema lineare di matrice

$$A = \begin{bmatrix} 5 & -1 & 0 & -1 \\ -1 & 5 & -1 & 0 \\ 0 & -1 & 5 & -1 \\ -1 & 0 & -1 & 5 \end{bmatrix}$$

e soluzione $\mathbf{x} = \begin{bmatrix} -1 \\ 2 \\ 0 \\ 1 \end{bmatrix}$.

- Verificare che la matrice A è simmetrica definita positiva.

Gradiente

```
function [x,nit]=gradiente(A,b,x0,nitmax,toll)
x=x0;
for nit=1:nitmax
    r=b-A*x;
    if norm(r)< toll*norm(b), return, end
    alpha=(r'*r)/(r'*A*r);
    x=x+alpha*r;
end
```

Equazioni no lineari

- Scrivere una funzione di Matlab che implementi il metodo delle secanti per l'approssimazione della soluzione di un'equazione non lineare $f(\alpha) = 0$:

  x^{-1} e x^0 assegnati
  per $k \geq 0$

  $$x^{k+1} = x^k - \frac{x^k - x^{k-1}}{f(x^k) - f(x^{k-1})} f(x^k)$$

  Usare il test d'arresto basato sull'incremento:

  $$\text{se } |x^{k+1} - x^k| < \texttt{toll} \quad \text{STOP.}$$

- Usando il metodo delle secanti risolvere l'equazione non lineare

  $$\alpha^3 - 7 = 0\,.$$

Secanti

```
function [xn,nit]=secanti(f,xo,x,nitmax,toll)
fxo=feval(f,xo);
for nit=1:nitmax
    fx=feval(f,x);
    xn=x-(x-xo)/(fx-fxo)*fx;
    if abs(x-xn)<toll, return, end
    xo=x;
    x=xn;
    fxo=fx;
end
```

Equazioni differenziali

- Scrivere una funzione di Matlab che implementi il metodo predictor-corrector di Milne-Simpson per approssimare la soluzione di un problema di Cauchy

$$u^*_{i+1} = u_{i-3} + \frac{4h}{3}[2f_i - f_{i-1} + 2f_{i-2}]$$
$$u_{i+1} = u_{i-1} + \frac{h}{3}[f(t_{i+1}, u^*_{i+1}) + 4f_i + f_{i-1}]\,.$$

Usare il metodo di Runge-Kutta 4

$$K_1 = f(t_i, u_i)$$
$$K_2 = f(t_i + \tfrac{h}{2}, u_i + \tfrac{h}{2}K_1)$$
$$K_3 = f(t_i + \tfrac{h}{2}, u_i + \tfrac{h}{2}K_2)$$
$$K_4 = f(t_i + h, u_i + hK_3)$$
$$u_{i+1} = u_i + \tfrac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

per inizializzare il metodo.

Equazioni differenziali

- Stimare esperimentalmente l'ordine di convergenza del metodo risolvendo il problema di Cauchy

$$\begin{aligned} y' &= te^{-y} \quad t \in [1,3] \\ y(1) &= 0 \end{aligned}$$

  Soluzione; $y(t) = log\left(\frac{t^2+1}{2}\right)$.
- Disegnare il grafico della funzione soluzione del problema di Cauchy e i punti della soluzione approssimata.

Milne-Simpson

```
function [t,u]=MilneSimpson(fun,t0,y0,T,n)
h=T/n;
t=[t0:h:t0+T];
u(1)=y0;
for i=1:3
    f(i)=feval(fun,t(i),u(i));
    K1=f(i);
    K2=feval(fun,t(i)+h/2,u(i)+h/2*K1);
    K3=feval(fun,t(i)+h/2,u(i)+h/2*K2);
    K4=feval(fun,t(i)+h,u(i)+h*K3);
    u(i+1)=u(i)+h/6*(K1+2*K2+2*K3+K4);
end
for i=4:n
    f(i)=feval(fun,t(i),u(i));
    aux=u(i-3)+4*h/3*(2*f(i)-f(i-1)+2*f(i-2));
    faux=feval(fun,t(i+1),aux);
    u(i+1)=u(i-1)+h/3*(faux+4*f(i)+f(i-1));
end
```