- Non si possono consultare libri, note, ed ogni altro materiale o persone durante l'esame ad eccezione delle funzioni Matlab fornite.
- Risolvere i seguenti esercizi con l'ausilio di Matlab.
- La durata del compito è di 90 minuti.
- Questo esame ha 3 domande, per un totale di 30/30 punti.
- Svolgere gli esercizi su fogli protocollo, indicando: nome, cognome, codice persona e data
- Per ciascun esercizio consegnare su webeep un file nominato, ad esempio, "esercizio1.m" con il codice Matlab sviluppato.
- Per utilizzare le funzioni Matlab sviluppate durante il corso, è necessario aggiungere la cartella con il comando `addpath functions2022`.

Esercizio 1 (punti 10)

Si consideri il seguente problema: determinare $\boldsymbol{x} \in \mathbb{R}^n$ tale che

$$A\boldsymbol{x} = \boldsymbol{b}$$

dove $A \in \mathbb{R}^{n \times n}$ è la matrice dei coefficienti di Hilbert di ordine n. In Matlab tale matrice si può costruire con il comando `hilb`. Il vettore $\boldsymbol{b} \in \mathbb{R}^n$ è tale che la soluzione è data da il vettore $x_i = 1$ per ogni $i = 1, \ldots, n$.

(a) (3 punti) Si calcoli la soluzione del problema per $n = [5, 10, 20]$ utilizzando la fattorizzazione LU e i metodi di sostituzione in avanti e indietro. Per ciascun valore di n, si riporti il numero di condizionamento della matrice e l'errore commesso. [M]

Soluzione. Ecco un possibile script

```
clc
close all
clear all

addpath functions2021

err = [];
condA = [];
for n = [5, 10, 20]
    A = hilb(n);
```

```
    condA = [condA, cond(A)];
    b = A*ones(n, 1);

    [L, U, P] = lu(A);
    y = fwsub(L, P*b);
    x = bksub(U, y);
    err = [err, norm(x - ones(n, 1))];
end
condA
err
```

Il condizionamento della matrice e gli errori commessi sono i seguenti

```
condA =

    4.7661e+05  1.6025e+13  2.1065e+18

err =

    2.7053e-11  5.2794e-05  150.7765
```

(b) (3 punti) Si introduca il concetto di condizionamento di una matrice e se ne discuta l'importanza sulla stabilità della risoluzione di un sistema lineare con il metodo di eliminazione di Gauss. Si commentino i risultati ottenuti al punto precedente. [T]

Soluzione. Vogliamo capire quanto dipende la soluzione calcolata $\boldsymbol{x}$ da possibili perturbazioni dei dati. Vorremmo calcolare il cosiddetto sistema originale invece risolviamo un sistema perturbato, ovvero

$$A\boldsymbol{x} = \boldsymbol{b} \quad \xrightarrow{\text{in realtà risolvo}} \quad (A + \delta A)(\boldsymbol{x} + \delta\boldsymbol{x}) = \boldsymbol{b} + \delta\boldsymbol{b},$$

dove δA è una matrice che perturba la matrice originale A, analogamente $\delta\boldsymbol{b}$ perturba il termine noto originale $\boldsymbol{b}$ e $\boldsymbol{x} + \delta\boldsymbol{x}$ è la soluzione del problema perturbato data dalla soluzione del problema originale e da un contributo $\delta\boldsymbol{x}$ che rappresenta la perturbazione.

Nell'ipotesi in cui l'errore commesso è solo al termine noto, ovvero se $\delta A = 0$, allora l'errore relativo che viene commesso nella risoluzione del problema con il metodo di eliminazione di Gauss è dato da

$$\frac{\|\delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \text{cond}(A)\frac{\|\delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|},$$

dove $\text{cond}(A)$ è il numero di condizionamento della matrice A che, per matrici

simmetriche e definite positive, è calcolabile come

$$\text{cond}(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}.$$

Diciamo che A è ben condizionata se $\text{cond}(A) \approx 1$, mentre A è mal condizionata se $\text{cond}(A) \gg 1$. In quest'ultimo caso, possiamo concludere che a piccole perturbazioni del termine noto, ovvero per $\|\delta \boldsymbol{b}\|$ piccolo, si possono avere grandi errori sulla soluzione, ovvero $\|\delta \boldsymbol{x}\|$ grande. Il condizionamento di una matrice è quindi un elemento fondamentale per poter valutare l'accuratezza della soluzione numerica ottenuta.

(c) (4 punti) Si ripeta il primo punto utilizzando il metodo del gradiente coniugato. Si imposti come massimo numero di iterazioni 100 e come tolleranza 10^{-12}. Si riporti l'errore ottenuto e il numero di iterazioni effettuate. Si commentino i risultati ottenuti in base alla teoria, sapendo che la condizione di convergenza nella funzione `gradconj` è basata sul residuo. [M+T]

Soluzione. Ecco un possibile script

```
%
err = [];
iterA = [];
for n = [5, 10, 20]
    A = hilb(n);
    b = A*ones(n, 1);

    [x, iter] = gradconj(A,b,zeros(n, 1),1e2,1e-12);
    iterA = [iterA, iter];
    err = [err, norm(x - ones(n, 1))];
end
iterA
err
```

Il numero di iterazioni effettuate e l'errore calcolato sono le seguenti

```
iterA =

     7    13    18

err =

    2.6983e-12 1.6026e-04 1.0743e-04
```

Sia $A \in \mathbb{R}^{n\times n}$ una matrice simmetrica e definita positiva, allora il metodo del gradiente coniugato converge alla soluzione del sistema $A\boldsymbol{x} = \boldsymbol{b}$, per ogni $\boldsymbol{x}^0 \in \mathbb{R}^n$ in al più n iterazioni. Inoltre, vale la stima

$$\left\|\boldsymbol{e}^k\right\|_A \leq \frac{2c^k}{1+c^{2k}} \left\|\boldsymbol{e}^0\right\|_A \quad \text{dove} \quad c = \frac{\sqrt{\text{cond}(A)}-1}{\sqrt{\text{cond}(A)}+1}.$$

Il numero di iterazioni, rispetto alla dimensione delle matrici, risulta piuttosto alto e vicino ad n sintomo appunto che il condizionamento delle matrici è molto alto.

Esercizio 2 (punti 10)

Si vuole interpolare la funzione f così definita

$$f(x) = \frac{e^{\cos(\pi x)}}{e^{-x} + \sin(\pi x)}$$

sull'intervallo $[-1, 1]$.

(a) (3 punti) Si calcolino le interpolanti Lagrangiane di grado $n = 5, 10, 20$ della funzione f su nodi equispaziati nell'intervallo $[-1, 1]$. Si rappresenti il grafico della funzione e delle interpolanti Lagrangiane e si valuti, per ciascun grado, l'errore di interpolazione in norma infinito sull'intervallo considerato. [M]

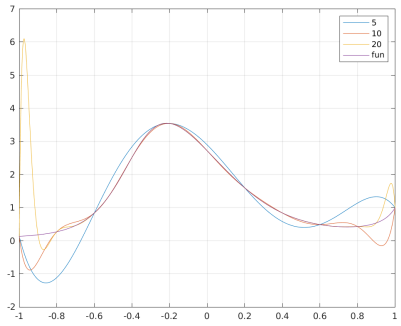
Soluzione. Ecco un possibile script

```
close all
clear all
clc

a = -1;
b = 1;
x_val = linspace(a, b, 1000);

f = @(x) (exp(cos(pi*x)))./(exp(-x) + sin(pi*x));
f_val = f(x_val);

err = [];
N = [5, 10, 20];
for n = N
    x_nod = linspace(a, b, n+1);
    poly = polyfit(x_nod, f(x_nod), n);
    poly_val = polyval(poly, x_val);
    err = [err max(abs(poly_val - f_val))];

    plot(x_val, poly_val)
    hold on
end
err

plot(x_val, f_val)
legend("5", "10", "20", "fun")
grid on
saveas(gcf, "es2_lagr.png")
```

L'errore ottenuto all'aumentare del grado di interpolazione è il seguente

```
err =

    1.4864    1.0402    5.9559
```

La soluzione ottenuta è riportata nel seguente grafico



(b) (3 punti) Si ripeta il punto precedente considerando le interpolanti Lagrangiane di grado $n = 5, 10, 20$ su nodi di Chebyshev–Gauss–Lobatto. [M]

Soluzione. Ecco un possibile script

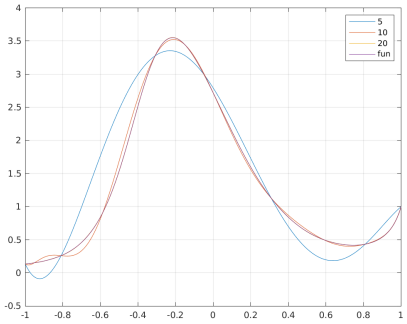
```
figure
err = [];
for n = N
    k = [0:n];
    t = - cos(pi*k/n);
    x_nod = (a+b)/2 + ((b-a)/2)*t;
    poly = polyfit(x_nod, f(x_nod), n);
    poly_val = polyval(poly, x_val);
    err = [err max(abs(poly_val - f_val))];

    plot(x_val, poly_val)
    hold on
end
err

plot(x_val, f_val)
legend("5", "10", "20", "fun")
grid on
saveas(gcf, "es2_cgl.png")
```

L'errore ottenuto all'aumentare del grado di interpolazione è il seguente

```
err =
```

```
    0.9896      0.1697      0.0044
```

La soluzione ottenuta è riportata nel seguente grafico



(c) (4 punti) Si riportino le proprietà di convergenza dell'interpolante Lagrangiana e si commentino i risultati ottenuti rispetto alla teoria. [T]

Soluzione. In generale non è detto che vale il seguente

$$\lim_{n\to\infty} \max_{x\in I} |E_n f(x)| = \lim_{n\to\infty} \max_{x\in I} |f(x) - \pi_n(x)| \to 0$$

essendo π_n un polinomio di grado n che interpola la funzione f. Se consideriamo infatti per l'interpolazione Lagrangiana

$$\left\| \Pi_n f - \Pi_n \widetilde{f} \right\|_\infty \leq \max_{i=0,\dots,n} \left| f(x_i) - \widetilde{f}(x_i) \right| \max_{x\in I} \left| \sum_{i=0}^{n} \mathcal{L}_i(x) \right|$$

L'ultimo termine non dipendete da f ma solo dai valori dei $\mathcal{L}_i$ nei nodi x_i, è detta costante di Lebesgue che, per nodi equispaziati, è data da

$$\Lambda_n = \max_{x\in I} \left| \sum_{i=0}^{n} \mathcal{L}_i(x) \right| \approx \frac{2^{n+1}}{en\log(n+\gamma)}$$

ed tende a diventare un numero molto grande per n crescente. Tuttavia, il valore di Λ_n dipende dal posizionamento dei nodi e se scegliessi i nodi detti di Chebychev-Gauss-Lobatto la costante di Lebesgue non crescerebbe così velocemente ottenendo quindi un'interpolazione stabile.

I grafici riportati nei punti precedenti mostrano esattamente questo fenomeno.

Esercizio 3 (punti 10)

Si consideri il seguente problema differenziale non lineare

$$\begin{cases} mx''(t) = -F(x(t)) \\ x'(0) = 1 \qquad\qquad\qquad\qquad t \in [0, T) \\ x(0) = 0 \end{cases}$$

dove $x(t)$ è la posizione tempo dipendente e F è una forza non lineare che assume la seguente forma

$$F(x) = \alpha x - \beta x^3,$$

assumiamo i seguenti valori per i dati del problema $m = 1$, $\alpha = 5$, $\beta = \pi/12$ e $T = 4$.

(a) (5 punti) Si riscriva il problema come un sistema differenziale del prim'ordine. Si applichi il metodo di Eulero in avanti per risolvere il problema per un numero di intervalli pari a $N = [5, 20, 80]$. [T+M]

Soluzione. Per trasformare il problema del second'ordine in un sistema del prim'ordine possiamo porre

$$y_0 = x \quad \text{e} \quad y_1 = x'$$

possiamo quindi riscrivere il problema nel seguente modo

$$\begin{cases} y_0' = y_1 \\ y_1' = -\frac{1}{m}\left(\alpha y_0 - \beta y_0^3\right) \end{cases}$$

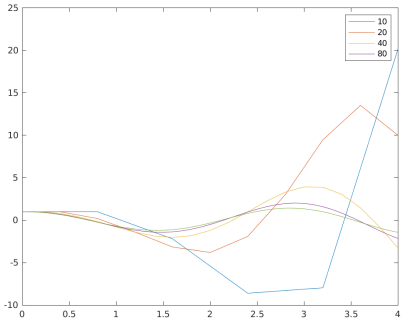
Ecco un possibile script

```
close all
clear all
clc

addpath functions2021

m = 1;
alpha = 5;
beta = 0*pi/12;
T = 4;

N = [5, 10, 20, 40, 80];

u_0 = [1; 0];
F = @(t, u) [u(2); -1/m*(alpha * u(1) - beta * u(1)^2)
   ];
```

```
for n = N
    [t, u] = eulero_avanti(F, 0, T, u_0, T / n);
    plot(t, u(1, :))
    hold on
end

legend("10", "20", "40", "80")
saveas(gcf, "es3_ee.png")
```

(b) (3 punti) Si rappresenti graficamente la soluzione ottenuta al punto precedente e si discuta il fenomeno che si osserva alla luce della teoria. [T]

Soluzione. La soluzione ottenuta è riportata nel seguente grafico



L'assoluta stabilità di uno schema numerico per la risoluzione di un problema di Cauchy è un concetto di stabilità su intervalli illimitati. Il metodo di Eulero in avanti risulta assolutamente stabile solo se prendo h sufficientemente piccolo, il metodo viene detto condizionatamente assolutamente stabile e deve soddisfare la seguente condizione

$$h < \frac{2}{|\lambda|}$$

(c) (2 punti) Si linearizzi il problema considerando $\beta = 0$ e si scriva il metodo di Eulero all'indietro per approssimare il problema. Si risolva il problema per un numero di intervalli pari a $N = [5, 20, 80]$. Cosa si osserva in questo caso? [T+M]

Soluzione. Considerando la formulazione al prim'ordine riportata precedente-

mente, ponendo $\beta = 0$ otteniamo

$$\begin{cases} y_0' = y_1 \\ y_1' = -\frac{\alpha}{m} y_0 \end{cases}$$

che scritto in forma vettoriale risulta

$$\boldsymbol{y}' = M\boldsymbol{y} \quad \text{dove} \quad M = \begin{bmatrix} 0 & 1 \\ -\frac{\alpha}{m} & 0 \end{bmatrix}$$

Utilizzando il metodo di Eulero implicito il precedente problema è approssimato nel seguente modo

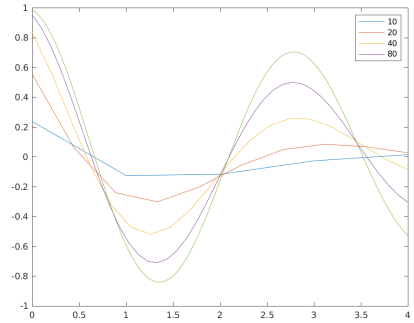
$$\begin{aligned} \boldsymbol{y}^{n+1} - \boldsymbol{y}^n &= \Delta t M \boldsymbol{y}^{n+1} \\ (I - \Delta t M)\boldsymbol{y}^{n+1} &= \boldsymbol{y}^n \end{aligned}$$

ad ogni istante temporale dobbiamo quindi risolvere il precedente sistema lineare.

Ecco un possibile script

```
%%

figure
for n = N
    delta_t = T / n;
    F = [[1, -delta_t]; [delta_t*alpha/m, 1]];
    u = u_0;
    x = [];
    for t = 1:n
        u = F \ u;
        x = [x, u(1)];
    end
    time = linspace(0, T, n);
    plot(time, x)
    hold on
end

legend("10", "20", "40", "80")
saveas(gcf, "es3_ei.png")
```

La soluzione ottenuta è riportata nel seguente grafico



Il metodo è incondizionatamente assolutamente stabile, non presenta quindi oscillazioni anche facendo pochi passi temporali.