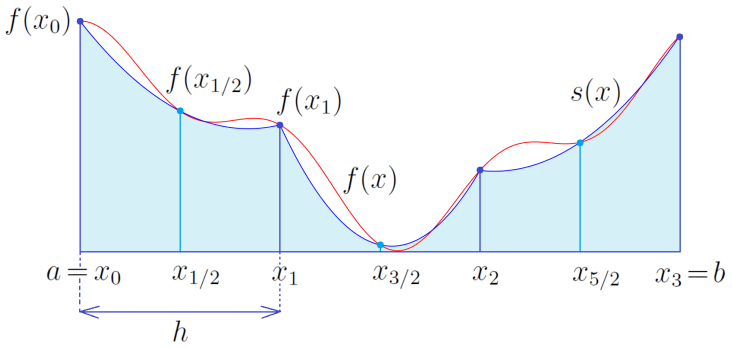

Figura 1.1: Illustrazione della formula di Cavalieri-Simpson per $n = 3$.

1 Esercizi

Esercizio 1.1. Esercizio d'implementazione dell'algoritmo di valutazione del polinomio d'interpolazione in più punti (Esercizio 1.11 sulle dispense).

Esercizio 1.2. Esercizio d'implementazione della formula dei trapezi (Esercizio 2.2 sulle dispense).

Esercizio 1.3. È data una funzione (integrabile) $f : [a, b] \to \mathbb{R}$ e si vuole calcolare un'approssimazione di $\int_a^b f(x)\mathrm{d}x$. A tal fine si suddivide l'intervallo $[a, b]$ in $n \geq 1$ sottointervalli tutti della stessa ampiezza $h = \frac{b-a}{n}$, si pone $x_j = a + jh$ per ogni $j = 0, 1, \ldots, n$, e si definisono i punti medi dei sottointervalli $x_{j+1/2} = \frac{1}{2}(x_j + x_{j+1})$ per $j = 0, \ldots, n-1$. Il valore che si prende come approssimazione di $\int_a^b f(x)\mathrm{d}x$ è $\int_a^b s(x)\mathrm{d}x$, dove

$$s : [a, b] \to \mathbb{R}, \quad \begin{cases} s(x) = f(x_j)\dfrac{(x - x_{j+1/2})(x - x_{j+1})}{(x_j - x_{j+1/2})(x_j - x_{j+1})} \\ \qquad + f(x_{j+1/2})\dfrac{(x - x_j)(x - x_{j+1})}{(x_{j+1/2} - x_j)(x_{j+1/2} - x_{j+1})} \\ \qquad + f(x_{j+1})\dfrac{(x - x_j)(x - x_{j+1/2})}{(x_{j+1} - x_j)(x_{j+1} - x_{j+1/2})}, \\ \text{per } x \in [x_j, x_{j+1}], \quad j = 0, \ldots, n-1, \end{cases}$$

è la funzione mostrata in Figura 1.1, che su ogni sottointervallo $[x_j, x_{j+1}]$ coincide con il polinomio (parabola) d'interpolazione di $f(x)$ sui nodi x_j, $x_{j+1/2}$, x_{j+1}. Quindi il valore che si prende come approssimazione di $\int_a^b f(x)\mathrm{d}x$ è

$$\begin{aligned} S_n &= \int_a^b s(x)\mathrm{d}x = \sum_{j=0}^{n-1} \int_{x_j}^{x_{j+1}} s(x)\mathrm{d}x \\ &= \sum_{j=0}^{n-1} \int_{x_j}^{x_{j+1}} \left[f(x_j)\frac{(x - x_{j+1/2})(x - x_{j+1})}{(x_j - x_{j+1/2})(x_j - x_{j+1})} \right. \\ &\qquad + f(x_{j+1/2})\frac{(x - x_j)(x - x_{j+1})}{(x_{j+1/2} - x_j)(x_{j+1/2} - x_{j+1})} \\ &\qquad \left. + f(x_{j+1})\frac{(x - x_j)(x - x_{j+1/2})}{(x_{j+1} - x_j)(x_{j+1} - x_{j+1/2})} \right] \mathrm{d}x \end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=0}^{n-1} \left[\frac{f(x_j)}{(x_j - x_{j+1/2})(x_j - x_{j+1})} \int_{x_j}^{x_{j+1}} (x - x_{j+1/2})(x - x_{j+1}) \mathrm{d}x \right. \\
&\qquad + \frac{f(x_{j+1/2})}{(x_{j+1/2} - x_j)(x_{j+1/2} - x_{j+1})} \int_{x_j}^{x_{j+1}} (x - x_j)(x - x_{j+1}) \mathrm{d}x \\
&\qquad \left. + \frac{f(x_{j+1})}{(x_{j+1} - x_j)(x_{j+1} - x_{j+1/2})} \int_{x_j}^{x_{j+1}} (x - x_j)(x - x_{j+1/2}) \mathrm{d}x \right] \\
&= \sum_{j=0}^{n-1} \left[\frac{f(x_j)}{(-h/2)(-h)} \cdot \frac{h^3}{12} + \frac{f(x_{j+1/2})}{(h/2)(-h/2)} \cdot \frac{-h^3}{6} + \frac{f(x_{j+1})}{h(h/2)} \cdot \frac{h^3}{12} \right] \quad \text{(verificare per esercizio quest'uguaglianza calcolando i tre integrali precedenti)} \\
&= \sum_{j=0}^{n-1} \left[f(x_j) \cdot \frac{h}{6} + f(x_{j+1/2}) \cdot \frac{4h}{6} + f(x_{j+1}) \cdot \frac{h}{6} \right] \\
&= \frac{h}{6} \left[f(x_0) + 4f(x_{1/2}) + f(x_1) + f(x_1) + 4f(x_{3/2}) + f(x_2) + f(x_2) + f(x_{5/2}) + f(x_3) + \dots \right. \\
&\qquad \left. + f(x_{n-1}) + 4f(x_{n-1/2}) + f(x_n) \right] \\
&= \frac{h}{6} \left[f(a) + f(b) + 2 \sum_{j=1}^{n-1} f(x_j) + 4 \sum_{j=0}^{n-1} f(x_{j+1/2}) \right]
\end{aligned}
$$

cioè

$$
\boxed{S_n = \frac{h}{6} \left[f(a) + f(b) + 2 \sum_{j=1}^{n-1} f(x_j) + 4 \sum_{j=0}^{n-1} f(x_{j+1/2}) \right]} \tag{1.1}
$$

La (1.1) prende il nome di *formula di Cavalieri-Simpson di ordine n*. L'ampiezza $h = \frac{b-a}{n}$ si chiama anche *passo (di discretizzazione) della formula S_n*.

Scrivere un programma MATLAB che implementa la formula di Cavalieri-Simpson. Il programma deve:
- prendere in input gli estremi a, b di un intervallo, una funzione $f(x)$ definita su $[a, b]$ e il numero $n \geq 1$ di sottointervalli in cui viene suddiviso $[a, b]$;
- restituire in output S_n, l'approssimazione di $\int_a^b f(x)\mathrm{d}x$ data dalla formula di Cavalieri-Simpson di ordine n.

Esercizio 1.4. Dato un sistema lineare $A\mathbf{x} = \mathbf{b}$ con $A \in \mathbb{C}^{n \times n}$ invertibile, si può costruire un metodo iterativo per risolvere tale sistema mediante la seguente procedura canonica: si considera una decomposizione di A del tipo $A = M - (M - A)$, dove $M \in \mathbb{C}^{n \times n}$ è una matrice invertibile detta *precondizionatore*, e si definisce il metodo

$$
\boxed{\begin{aligned}
&\mathbf{x}^{(0)} \in \mathbb{C}^n \text{ dato,} \\
&\mathbf{x}^{(k+1)} = M^{-1}(M - A)\mathbf{x}^{(k)} + M^{-1}\mathbf{b} = \mathbf{x}^{(k)} + M^{-1}\mathbf{r}^{(k)}, \quad k = 0, 1, 2, \dots
\end{aligned}} \tag{1.2}
$$

dove $\mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)}$ è il residuo del sistema al passo k. Il metodo di Gauss-Seidel con rilassamento, noto anche come metodo SOR, è il metodo della forma (1.2) in cui il precondizionatore M è dato da $M = \frac{1}{\omega}D + E - D$, dove D è la parte diagonale di A come nel metodo di Jacobi, E è la parte triangolare inferiore di A come nel metodo di Gauss-Seidel, e $\omega \neq 0$ è un parametro scelto dall'utente, detto parametro di rilassamento. Osserviamo che $M = \frac{1}{\omega}D + E - D$ è invertibile se e solo se D è invertibile, per cui il metodo SOR è applicabile se e solo se D è invertibile, cioè se e solo se tutti gli elementi diagonali di A sono diversi da 0. Osserviamo inoltre che per $\omega = 1$ si ottiene il metodo di Gauss-Seidel classico.

Scrivere un programma MATLAB che implementa il metodo SOR. Il programma deve:

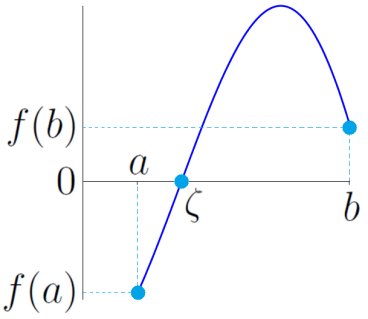

Figura 1.2: Una funzione continua $f : [a, b] \to \mathbb{R}$ tale che $f(a)f(b) < 0$ possiede almeno uno zero $\zeta \in (a, b)$.

- prendere in input la matrice A e il termine noto $\mathbf{b}$ del sistema lineare da risolvere $A\mathbf{x} = \mathbf{b}$, un parametro di rilassamento $\omega \neq 0$, una soglia di precisione ε, un vettore d'innesco $\mathbf{x}^{(0)}$ e un numero massimo d'iterazioni consentite $N_{\max}$;
- restituire in output il primo vettore $\mathbf{x}^{(K)}$ calcolato dal metodo SOR (con $0 \leq K \leq N_{\max}$) che soddisfa la condizione di arresto del residuo $\|\mathbf{r}^{(K)}\|_2 \leq \varepsilon\|\mathbf{b}\|_2$, il relativo indice K che conta il numero d'iterazioni effettuate, e la norma $\|\mathbf{r}^{(K)}\|_2$ del residuo a cui ci si arresta. Se nessuno dei vettori $\mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(N_{\max})}$ soddisfa la condizione di arresto del residuo allora il programma deve restituire in output $\mathbf{x}^{(N_{\max})}$, il relativo indice $N_{\max}$, e la norma $\|\mathbf{r}^{(N_{\max})}\|_2$ dell'ultimo residuo.

Esercizio 1.5. Sia $f : [a, b] \to \mathbb{R}$ una funzione continua su $[a, b]$ tale che $f(a)$ e $f(b)$ hanno segno opposto: $f(a)f(b) < 0$. Un teorema dell'analisi matematica (teorema degli zeri) garantisce che la funzione $f(x)$ ha almeno uno zero nell'intervallo (a, b), cioè esiste un punto $\zeta \in (a, b)$ tale che $f(\zeta) = 0$; si veda la Figura 1.2. Supponiamo che $f(x)$ abbia un unico zero ζ in (a, b). Un metodo per determinare un'approssimazione ξ di ζ è il metodo di bisezione: fissata una soglia di precisione $\varepsilon > 0$, il metodo costruisce la successione di intervalli

$$[\alpha_k, \beta_k], \qquad k = 0, 1, 2, \ldots$$

in cui $[\alpha_0, \beta_0] = [a, b]$ e, per $k \geq 1$,

$$[\alpha_k, \beta_k] = \begin{cases} \left[\alpha_{k-1}, \dfrac{\alpha_{k-1} + \beta_{k-1}}{2}\right], & \text{se } \zeta \in \left[\alpha_{k-1}, \dfrac{\alpha_{k-1} + \beta_{k-1}}{2}\right] \text{ cioè } f(\alpha_{k-1}) f\Big(\dfrac{\alpha_{k-1} + \beta_{k-1}}{2}\Big) \leq 0, \\ \left[\dfrac{\alpha_{k-1} + \beta_{k-1}}{2}, \beta_{k-1}\right], & \text{altrimenti.} \end{cases}$$

La successione di intervalli così costruita gode delle seguenti proprietà:
- $\zeta \in [\alpha_k, \beta_k]$ per tutti i $k \geq 0$;
- ogni intervallo è metà del precedente e dunque la lunghezza di $[\alpha_k, \beta_k]$ è $\beta_k - \alpha_k = \frac{b-a}{2^k}$ per ogni $k \geq 0$.

Il metodo si arresta al primo indice K tale che $\beta_K - \alpha_K \leq \varepsilon$ e restituisce come risultato il punto medio ξ dell'intervallo $[\alpha_K, \beta_K]$ dato da $\xi = \frac{\alpha_K + \beta_K}{2}$. In questo modo, siccome $\zeta \in [\alpha_K, \beta_K]$, si ha $|\xi - \zeta| \leq \frac{\varepsilon}{2}$. Osserviamo che l'indice di arresto K è il più piccolo intero ≥ 0 tale che

$$\beta_K - \alpha_K \leq \varepsilon \quad \Longleftrightarrow \quad \frac{b-a}{2^K} \leq \varepsilon \quad \Longleftrightarrow \quad 2^K \geq \frac{b-a}{\varepsilon} \quad \Longleftrightarrow \quad K \geq \log_2\Big(\frac{b-a}{\varepsilon}\Big),$$

cioè $K = \left\lceil \log_2\left(\frac{b-a}{\varepsilon}\right) \right\rceil$.

Scrivere un programma MATLAB che implementa il metodo di bisezione. Il programma deve:
- prendere in input gli estremi a, b di un intervallo, una funzione continua $f : [a, b] \to \mathbb{R}$, con $f(a)f(b) < 0$ e con un unico zero $\zeta \in (a, b)$, e un $\varepsilon > 0$;

- restituire in output l'approssimazione ξ di ζ ottenuta con il metodo di bisezione sopra descritto, l'indice di arresto K del metodo, e il valore $f(\xi)$ (che sarà all'incirca pari a $0 = f(\zeta)$).

Esercizio 1.6. Sia $g : [a, b] \to \mathbb{R}$ una funzione di classe $C^1[a, b]$ con $|g'(x)| < 1$ per ogni $x \in [a, b]$ e supponiamo che $\alpha = g(\alpha)$ per un qualche punto $\alpha \in [a, b]$.[1] In tal caso, si può dimostrare che:
1. α è l'unica soluzione dell'equazione $x = g(x)$ in $[a, b]$;
2. la successione

$$\boxed{\begin{array}{l} x_0 \in [a, b] \text{ dato,} \\ x_{k+1} = g(x_k), \quad k = 0, 1, 2, \ldots \end{array}} \tag{1.3}$$

converge ad α per "molte" scelte di $x_0 \in [a, b]$ e in particolare converge sicuramente ad α se x_0 è l'estremo dell'intervallo $[a, b]$ più vicino ad α.

Il metodo (1.3) permette quindi di calcolare un'approssimazione della soluzione α dell'equazione $x = g(x)$: basta scegliere un punto $x_0 \in [a, b]$ tale che la successione (1.3) converga ad α, e calcolare poi i termini della successione (1.3) arrestandosi a un indice K tale che x_K sia sufficientemente vicino ad α. Il metodo (1.3) prende il nome di *metodo d'iterazione funzionale* o *metodo di punto fisso* per risolvere l'equazione $x = g(x)$.

Scrivere un programma MATLAB che implementa il metodo (1.3). Il programma deve:
- prendere in input la funzione $g(x)$, una soglia di precisione ε, il punto d'innesco x_0, e un numero massimo d'iterazioni consentite $N_{\max}$;
- restituire in output il primo termine x_K calcolato dal metodo (1.3) (con $1 \le K \le N_{\max}$) che soddisfa la condizione di arresto $|x_K - x_{K-1}| \le \varepsilon$, il relativo indice K che conta il numero d'iterazioni effettuate, e il modulo dell'errore $|x_K - g(x_K)|$. Se nessuno dei termini $x_0, \ldots, x_{N_{\max}}$ soddisfa la condizione di arresto precedente, allora il programma deve restituire in output $x_{N_{\max}}$, il relativo indice $N_{\max}$, e il modulo dell'errore $|x_{N_{\max}} - g(x_{N_{\max}})|$.

2 Problemi

Problema 2.1. Si consideri la funzione $\sqrt{x}$.

(a) Sia $p(x)$ il polinomio d'interpolazione di $\sqrt{x}$ sui nodi

$$x_0 = 0, \quad x_1 = \frac{1}{64}, \quad x_2 = \frac{4}{64}, \quad x_3 = \frac{9}{64}, \quad x_4 = \frac{16}{64}, \quad x_5 = \frac{25}{64}, \quad x_6 = \frac{36}{64}, \quad x_7 = \frac{49}{64}, \quad x_8 = 1.$$

Calcolare il vettore (colonna)

$$\begin{bmatrix} p(\zeta_1) - \sqrt{\zeta_1} & p(\zeta_2) - \sqrt{\zeta_2} & \cdots & p(\zeta_{21}) - \sqrt{\zeta_{21}} \end{bmatrix}^T$$

dove $\zeta_i = \frac{i-1}{20}$ per $i = 1, \ldots, 21$, e osservare in che modo varia la differenza $p(\zeta_i) - \sqrt{\zeta_i}$ al variare di i da 1 a 21.

(b) Tracciare il grafico di $\sqrt{x}$ e di $p(x)$ sull'intervallo $[0, 1]$, ponendo i due grafici su un'unica figura e inserendo una legenda che ci dica qual è la funzione $\sqrt{x}$ e qual è il polinomio $p(x)$.

Problema 2.2. Si consideri la funzione

$$f(x) = \mathrm{e}^x.$$

Per ogni intero $n \ge 1$ indichiamo con I_n la formula dei trapezi di ordine n per approssimare

$$I = \int_0^1 f(x)\mathrm{d}x = 1.7182818284590...$$

[1] Si dice in tal caso che α è un *punto fisso* della funzione $g(x)$ in $[a, b]$, perché la funzione $g(x)$ "lascia fisso" α essendo $g(\alpha) = \alpha$.

(a) Per ogni fissato $\varepsilon > 0$ determinare un $n = n(\varepsilon)$ tale che $|I - I_n| \leq \varepsilon$.
(b) Costruire una tabella che riporti vicino ad ogni $\varepsilon \in \{10^{-1}, 10^{-2}, \ldots, 10^{-10}\}$:
- il numero $n(\varepsilon)$;
- il valore I_n per $n = n(\varepsilon)$;
- il valore esatto I (in modo da confrontarlo con I_n);
- l'errore $|I - I_n|$ (che deve essere $\leq \varepsilon$).

(c) Calcolare le approssimazioni di I ottenute con le formule dei trapezi I_2, I_4, I_8, I_{16} e confrontarle con il valore esatto I.
(d) Sia $p(x)$ il polinomio d'interpolazione dei valori I_2, I_4, I_8, I_{16} sui nodi h_2^2, h_4^2, h_8^2, h_{16}^2, dove $h_2 = \frac{1}{2}$, $h_4 = \frac{1}{4}$, $h_8 = \frac{1}{8}$, $h_{16} = \frac{1}{16}$ sono i passi di discretizzazione relativi alle formule dei trapezi I_2, I_4, I_8, I_{16} rispettivamente. Calcolare $p(0)$ e confrontare I_2, I_4, I_8, I_{16}, $p(0)$ con il valore esatto I. Che cosa si nota?

Problema 2.3. Consideriamo la funzione $f(x) = \frac{1}{x \log x}$ e indichiamo rispettivamente con I_n e S_n la formula dei trapezi e di Cavalieri-Simpson di ordine n per approssimare $I = \int_2^5 f(x)\mathrm{d}x$.
(a) Calcolare I prima manualmente e poi con la funzione simbolica `int` di MATLAB.
(b) Costruire una tabella che riporti vicino ad ogni valore di

$$n = 5, 10, 20, 40, 80, 160, 320, 640, 1280, 2560$$

sia le approssimazioni di I ottenute con I_n e S_n sia i relativi errori $|I_n - I|$ e $|S_n - I|$. Quale delle formule I_n e S_n converge più velocemente al valore esatto I al crescere di n?

Problema 2.4. Si consideri il sistema lineare $A\mathbf{x} = \mathbf{b}$ dove $\mathbf{b} = [1, 0, -2, 0]^T$ e

$$A = \begin{bmatrix} 1 & -\frac{1}{4} & \frac{1}{3} & 0 \\ -1 & 2 & 0 & \frac{1}{2} \\ 2 & 1 & 3 & -\frac{1}{3} \\ -1 & -2 & -4 & 7 \end{bmatrix}.$$

(a) Sia G_ω la matrice d'iterazione del metodo SOR con parametro di rilassamento $\omega > 0$ per risolvere il sistema dato. Tracciare con MATLAB il grafico della funzione $\omega \mapsto \rho(G_\omega)$ per $\omega \in (0, 2]$ e determinare il valore $\omega_{\text{opt}} \in \{\frac{i}{m} : i = 1, \ldots, 2m\}$ che minimizza $\rho(G_\omega)$ nel caso $m = 1000$.
(b) Calcolare $\rho(G_\omega)$ nel caso $\omega = 1$ e $\omega = \omega_{\text{opt}}$.
(c) Riportare in una tabella:
- le prime 10 iterazioni del metodo di Gauss-Seidel (classico) per risolvere il sistema dato, partendo dal vettore d'innesco $\mathbf{x}^{(0)} = [0, 0, 0, 0]^T$;
- le prime 10 iterazioni del metodo SOR con parametro di rilassamento ω_{opt} per risolvere il sistema dato, partendo dal vettore d'innesco $\mathbf{x}^{(0)} = [0, 0, 0, 0]^T$.

Confrontare le iterazioni con la soluzione esatta $\mathbf{x}$ del sistema dato calcolando in particolare la norma ∞ della differenza fra le iterazioni e la soluzione: quale dei due metodi converge più velocemente alla soluzione esatta?

Problema 2.5. Consideriamo i seguenti due casi:
- $f(x) = x^3 + 3x - 1 - \mathrm{e}^{-x^2}$ e $[a, b] = [0, 1]$;
- $f(x) = \cos x - x$ e $[a, b] = [0, \pi]$.

Per ciascuno di questi due casi, risolvere i seguenti punti.
(a) Verificare che $f(a)f(b) < 0$.
(b) Tracciare il grafico di $f(x)$ su $[a, b]$ e verificare che $f(x)$ ha un unico zero ζ nell'intervallo (a, b).

(c) Dimostrare analiticamente che $f(x)$ ha un'unico zero ζ nell'intervallo (a, b).
(d) Costruire una tabella che riporti vicino ad ogni $\varepsilon \in \{10^{-1}, 10^{-2}, \dots, 10^{-10}\}$:
- un'approssimazione ξ_ε di ζ, calcolata con il metodo di bisezione, che soddisfa $|\xi_\varepsilon - \zeta| \leq \varepsilon$;
- il numero d'iterazioni K_ε effettuate dal metodo di bisezione per calcolare l'approssimazione ξ_ε;
- il valore $f(\xi_\varepsilon)$.

Problema 2.6. Consideriamo le seguenti due funzioni e gli intervalli riportati a fianco di esse:
- $g(x) = \frac{1+\mathrm{e}^{-x^2}}{x^2+3}$, $\quad [a, b] = [0, 1]$;
- $g(x) = \cos x$, $\quad [a, b] = [0, \pi/3]$.

Per ciascuno di questi due casi, risolvere i seguenti punti.
(a) Tracciare il grafico di $y = g(x)$ e il grafico di $y = x$ sull'intervallo $[a, b]$ e dedurre che l'equazione $x = g(x)$ ha un'unica soluzione $\alpha \in [a, b]$. Che cosa rappresenta α nel grafico tracciato?
(b) Dimostrare analiticamente che l'equazione $x = g(x)$ ha un'unica soluzione nell'intervallo $[a, b]$.
(c) Costruire una tabella che riporti vicino ad ogni $\varepsilon \in \{10^{-1}, 10^{-2}, \dots, 10^{-10}\}$:
- un'approssimazione α_ε di α—calcolata con il metodo d'iterazione funzionale (1.3) innescato partendo da un punto $x_0 \in [a, b]$—che soddisfa la condizione di arresto del metodo (1.3) proposta nell'Esercizio 1.6 usando come soglia di precisione ε;
- il punto d'innesco $x_0 \in [a, b]$ utilizzato per innescare il metodo (1.3) e ottenere α_ε;
- il numero d'iterazioni K_ε effettuate dal metodo (1.3) per calcolare l'approssimazione α_ε;
- il valore $|\alpha_\varepsilon - g(\alpha_\varepsilon)|$.