La Classe dei Triangoli

`https://github.com/morganapederzoli/Triangoli_esame`

Morgana M. E. Pederzoli

August 2020

Indice

1 Introduzione

Il triangolo è il poligono che sta alla base di tutta la geometria euclidea.

Grazie alla caratteristica di *indeformabilità*, una qualsiasi terna numerica, che rispetti la disuguaglianza triangolare, determina univocamente un triangolo ed è sufficiente per trovare analiticamente tutte le caratteristiche e le proprietà del triangolo stesso.

Non è tuttavia questo l'unico modo di ottenere un triangolo, sappiamo infatti dai teoremi della geometria euclidea che i triangoli godono di tre principi di equivalenza:

– se due triangoli hanno tutti e tre i lati rispettivamente congruenti essi sono congruenti;
– se due triangoli hanno due lati congruenti e l'angolo tra essi compreso congruente allora sono congruenti;
– se due triangoli hanno un lato congruente e due angoli congruenti allora sono congruenti.

Esistono perciò tre tipologie di terne che identificano univocamente un triangolo:

– tre lati;
– due lati e un angolo (quello compreso);
– un lato e due angoli.

Lo scopo primario di questo programma C++ è quindi quello di creare, se possibile, triangoli a partire dai dati forniti dall'utente; in secondo luogo fornisce tutte le informazioni (area, perimetro, angoli e altezze relative ad ogni lato) riguardo al triangolo creato; infine è anche in grado di mostrare graficamente il triangolo, i suoi punti notevoli (baricentro, incentro, circocentro e ortocentro) e le circonferenze inscritta e circoscritta.

2 Istruzioni per l'uso

Questo programma C++ è pensato per essere eseguito su un OS Debian-based, ma può essere eseguito su una qualunque altra piattaforma che abbia installato un compilatore `gcc` almeno alla versione 7.5 e `gnuplot` almeno alla versione 5.2, a patto di piccole modifiche al codice. In particolare va modificata la parte di codice in cui si esegue `gnuplot`, che mentre per un OS Debian-based è semplicemente `"gnuplot"` per un sistema Windows, ad esempio, è `"start gnuplot"`.

Per scaricare comodamente il programma da GitHub occorre inoltre avere `git` installato. Per scaricare il programma aprire il terminale e scrivere:

```
$ git clone https://github.com/morganapederzoli/Triangoli_esame
```

A questo punto, se si dispone di `CMake` almeno alla versione 3.13, per compilare ed eseguire il programma è sufficiente eseguire da terminale:

```
$ cd Triangoli_esame/triangoli-master/Programma
$ cmake .
$ make
$ ./triangoli
```

Infatti nella cartella `Programma` è presente un file `CMakeLists.txt` che in automatico compila il programma con le opzioni "`-Wall -Wextra -fsanitize=address`" producendo un file eseguibile nominato `triangoli`.

Se non si dispone di `CMake` è sufficiente eseguire da terminale:

```
$ cd Triangoli_esame/triangoli-master/Programma
$ g++ -std=c++17 -Wall -Wextra -fsanitize=address -o triangoli
  triangoli.cpp
$ ./triangoli
```

3 Implementazione

3.1 Struttura generale

In linea con le coding convention più standard si è scelto di suddividere il file in

- header file `triangoli.h` per le dichiarazioni di tutti i metodi della classe, dei suoi attributi e delle funzioni esterne;
- file di implementazione `triangoli.cpp` in cui è presente anche il `main` del programma.

Nella trattazione a seguire, in cui sono elencati i principali componenti della classe `Triangoli`, viene sottinteso il *namespace* `std`.

class Triangoli

Attributi

vector<double> l
Attributo privato della classe, è l'insieme dei lati del triangolo

Costruttore

Triangoli(vector<double>, double)
Costruttore parametrico: prende un vector di 3 elementi e inizializza l'attributo l con i valori del vector ; per semplificare la parte grafica e la scelta del sistema di coordinate, il costruttore, prima di inizializzare l, mette in ordine decrescente gli elementi del vector in argomento.

Metodi

double perimetro()
Restituisce il perimetro del triangolo.

double area(double)
Restituisce l'area usando la formula di Erone (vedi appendice A), per questo motivo ha bisogno di ricevere il perimetro come argomento.

vector<double> angoli()
Restituisce un vector contenente gli angoli sfruttando il teorema di Carnot (*o teorema del coseno*, vedi appendice B.2), tali angoli sono rispettivamente opposti al lato con lo stesso indice nel vector l.

vector<double> altezze(double)
Restituisce le tre altezze del triangolo, tali altezze sono rispettive al lato con lo stesso indice nel vector l.

vector<vector<double>> coord(vector<double>, vector<double>)
Restituisce un vector bidimensionale contenente le coordinate di ciascuno dei vertici.

vector<double> bari(vector<vector<double>>)
Restituisce un vector con le coordinate del baricentro (punto di incontro delle mediane) del triangolo.

vector<double> in(vector<vector<double>>, double, double)
Restituisce un vector con le coordinate dell'incentro (punto di incontro delle bisettrici) del triangolo e il valore del raggio della circonferenza inscritta.

vector<double> orto(vector<vector<double>>)
Restituisce un vector con le coordinate dell'ortocentro (punto di incontro delle altezze) del triangolo.

vector<double> circo(vector<vector<double>>, double)
Restituisce un vector con le coordinate del circocentro (punto di incontro degli assi dei lati) del triangolo e il valore del raggio della circonferenza circoscritta.

vector<vector<double>> coord_not(vector<vector<double>>, double, double)
Raggruppa tutti i vector dei punti notevoli in un unico vector di vector per semplificare la trattazione nella parte grafica.

void disegnetto (vector<vector<double>>, vector<vector<double>>)
Fa eseguire il programma gnuplot per ottenere la rappresentazione grafica del triangolo e, a scelta dell'utente, anche dei punti notevoli e delle circonferenze inscritta e circoscritta.

Funzioni ausiliarie

template<class A> void lettura(A &)
Funzione `template` necessaria per evitare che l'utente inserisca in input da tastiera elementi di tipo diverso da quello richiesto. Si è scelto di usare una `template` perchè nel corso del programma gli input da fornire sono di tipi diversi.
void pulisci_cin()
Funzione che assicura che in `cin` non rimangano in memoria valori indesiderati (da un precedente input da tastiera).
vector<double> sinus(double, double,double)
Funzione che restituisce i tre lati del triangolo in un `vector` sfruttando il teorema del seno (vedi appendice B.1) e che prende in argomento un lato e due angoli.
vector<double> carnot(double, double,double)
Funzione che restituisce i tre lati del triangolo in un `vector` sfruttando il teorema del coseno (vedi appendice B.2). Come argomenti prende i due lati ed un angolo.
void conf_lat(vector<double>, bool)
Funzione che confronta i lati del triangolo e che comunica su terminale all'utente se il triangolo, eventualmente creato, è rettangolo, equilatero, isoscele o scaleno.
bool dis_tri(vector<double>)
Funzione che verifica che i tre valori contenuti nel `vector` che riceve in argomento soddisfino la disuguaglianza triangolare. Tale funzione restituisce 1 se la disuguaglianza è rispettata, altrimenti restituisce 0. N.B. *Per gli scopi del programma si è scelto di usare la disuguaglianza stretta.*

`int main ()`

Lo scheletro del `main` è così organizzato: prima vengono dichiarate le variabili usate, poi viene chiesto all'utente di scegliere quali elementi del triangolo fornire al programma. A seconda di questa scelta, si può entrare in uno dei tre differenti `case`.

```
//dichiarazioni
while(true){
    //...
    lettura(s);
    check=1;
    switch(s){
        case(1){        //Trilatula
            while(check){
            //...
            lettura(check);
            }
            break;
        }
        case(2){        //Duangula
            while(check){
            //...
            lettura(check);
            }
            break;
        }
        case(3){        //Dulatula
            while(check){
```

```
            //...
            lettura(check);
            }
            break;
        }
        case(0) return 0;
    }
}
```

Si noti la struttura a matrioska del `main`. La presenza dei `while()` all'interno dei `case` permette all'utente di riutilizzare la stessa procedura di inserimento dei dati finché lo desidera, infatti alla fine di ogni iterazione del `while()` viene chiesto esplicitamente se si vuole cambiare modalità di inserimento dati oppure no. La presenza del ciclo `while(true)` più esterno permette di usare l'intero programma illimitatamente fino a quando l'utente non decide di passare al `case(0)` che chiude il programma.

Analizziamo il ruolo dei 3 `case`.

`case(1)` Permette all'utente di verificare se una terna di numeri può essere un insieme di lati di un triangolo. In caso di esito positivo vi è la possibilità, decisa sempre dall'utente, di creare un oggetto della classe `Triangoli` e usarne i relativi metodi.

`case(2)` Permette all'utente di trovare immediatamente i lati del triangolo inequivocabilmente identificato da un lato e una coppia di angoli forniti dall'utente; successivamente vi è la possibilità di creare un oggetto della classe `Triangoli` e usarne i relativi metodi.

`case(3)` Permette all'utente di trovare immediatamente i lati del triangolo identificato univocamente da due lati e dall'angolo tra essi compreso, questi ultimi forniti ancora una volta come input del programma. Come in precedenza, l'utente può creare un oggetto della classe `Triangoli` e usarne i relativi metodi.

3.2 Parte grafica

Per poter ottenere la rappresentazione grafica del triangolo, dei suoi punti notevoli e delle circonferenze, si è ricorsi al programma `gnuplot`, già noto grazie ai corsi di laboratorio. Dato che `gnuplot` funziona attraverso input da terminale è stato sufficiente istanziare un oggetto della classe `stringstream` in cui inserire, tramite l'operatore `<<`, le linee di comando da far eseguire a `gnuplot`. La funzione `system(const char*)` serve, quindi, a passare la direttiva al processore di far eseguire a `gnuplot` tali linee di comando. Si noti che per lanciare `gnuplot` da terminale su un sistema Linux derivato da Debian è sufficiente digitare: `$ gnuplot`. Tale direttiva in Windows viene sostituita da: `$ start gnuplot`. Quindi, il programma deve essere modificato coerentemente per poter funzionare su Windows. Le modifiche da apportare sono tuttavia minime.

4 Esempio

Di seguito viene riportato un esempio di esecuzione del programma.

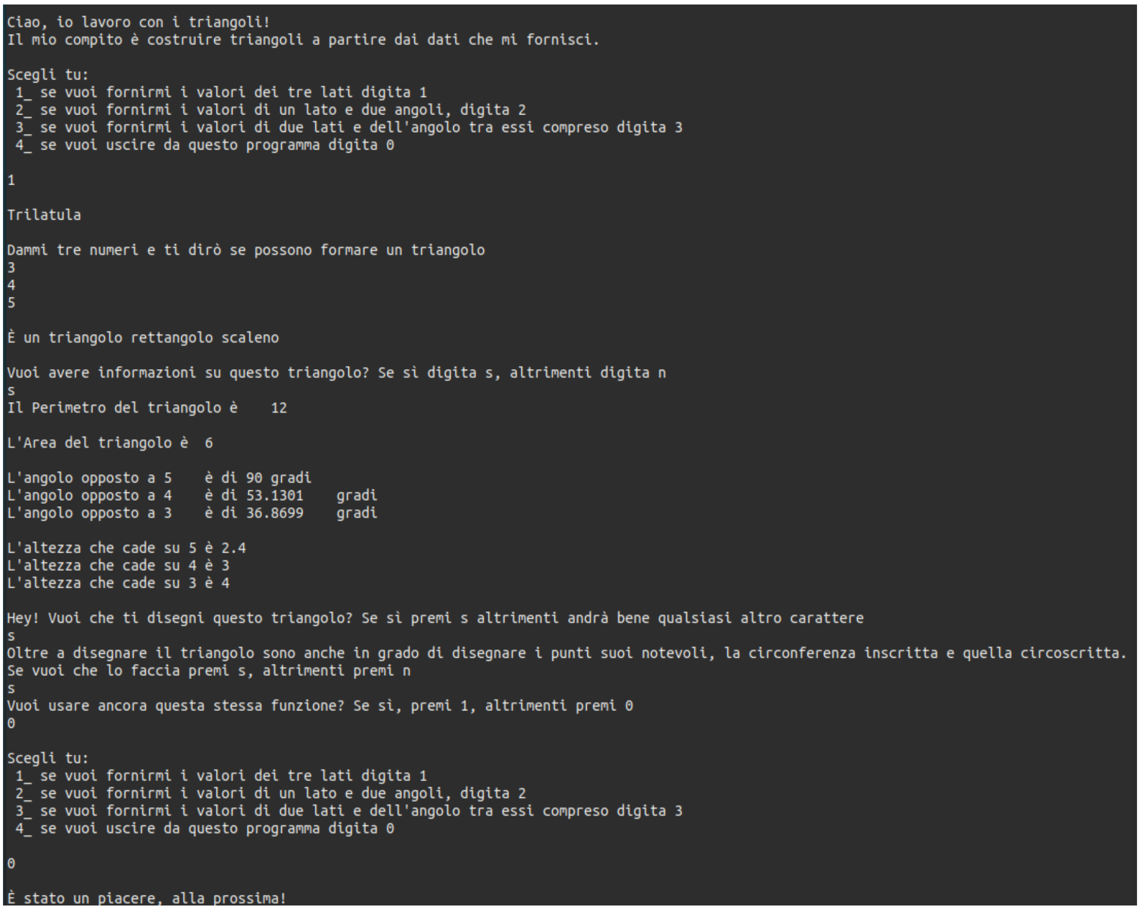
Viene fatta eseguire la funzione che costruisce il triangolo a partire dai tre lati.

Come input, viene data la terna pitagorica 3-4-5 (in unità arbitrarie, che possono essere centimetri, metri o altri multipli), ci si aspetta un triangolo rettangolo scaleno con angoli interni

$$\arcsin(3/5) \approx 36.87°,$$
$$\arcsin(4/5) \approx 53.13°,$$
$$90°.$$

Dalla geometria euclidea sappiamo che un triangolo rettangolo è sempre inscritto in una semicirconferenza con diametro pari all'ipotenusa perciò il circocentro deve corrispondere al punto medio dell'ipotenusa; essendo poi ogni cateto l'altezza rispettiva dell'altro, l'ortocentro deve coincidere col vertice corrispondente all'angolo retto. Si dimostra inoltre che le altezze aspettate sono rispettivamente di 4 u.a, 3 u.a e 2.4 u.a (vedi appendice D).

In figura 1 è riportato il dialogo con cui si presenta il programma. Dall'immagine si possono osservare i risultati ottenuti, comunque riportati in tabella 1.



```
Ciao, io lavoro con i triangoli!
Il mio compito è costruire triangoli a partire dai dati che mi fornisci.

Scegli tu:
 1_ se vuoi fornirmi i valori dei tre lati digita 1
 2_ se vuoi fornirmi i valori di un lato e due angoli, digita 2
 3_ se vuoi fornirmi i valori di due lati e dell'angolo tra essi compreso digita 3
 4_ se vuoi uscire da questo programma digita 0

1

Trilatula

Dammi tre numeri e ti dirò se possono formare un triangolo
3
4
5

È un triangolo rettangolo scaleno

Vuoi avere informazioni su questo triangolo? Se sì digita s, altrimenti digita n
s
Il Perimetro del triangolo è    12

L'Area del triangolo è  6

L'angolo opposto a 5    è di 90 gradi
L'angolo opposto a 4    è di 53.1301    gradi
L'angolo opposto a 3    è di 36.8699    gradi

L'altezza che cade su 5 è 2.4
L'altezza che cade su 4 è 3
L'altezza che cade su 3 è 4

Hey! Vuoi che ti disegni questo triangolo? Se sì premi s altrimenti andrà bene qualsiasi altro carattere
s
Oltre a disegnare il triangolo sono anche in grado di disegnare i punti suoi notevoli, la circonferenza inscritta e quella circoscritta.
Se vuoi che lo faccia premi s, altrimenti premi n
s
Vuoi usare ancora questa stessa funzione? Se sì, premi 1, altrimenti premi 0
0

Scegli tu:
 1_ se vuoi fornirmi i valori dei tre lati digita 1
 2_ se vuoi fornirmi i valori di un lato e due angoli, digita 2
 3_ se vuoi fornirmi i valori di due lati e dell'angolo tra essi compreso digita 3
 4_ se vuoi uscire da questo programma digita 0

0

È stato un piacere, alla prossima!
```

Figura 1: Dialogo del programma.

	Valore/i atteso/i	Valore/i ottenuto/i
Area	6 u.a^2	6 u.a^2
Perimetro	12 u.a	12 u.a
Angoli	36.87°, 53.13°, 90°	36.87°, 53.13°, 90°
Altezze	4, 3, 2.4	4, 3, 2.4

Tabella 1: Comparazione dei risultati ottenuti con quelli aspettati. Tutti i risultati sono espressi in unità arbitrarie u.a

Nelle figure 2 e 3 sono riportate le rappresentazioni grafiche che il programma produce sia del solo triangolo e che del triangolo con i suoi punti notevoli e le circonferenze inscritta e circoscritta.

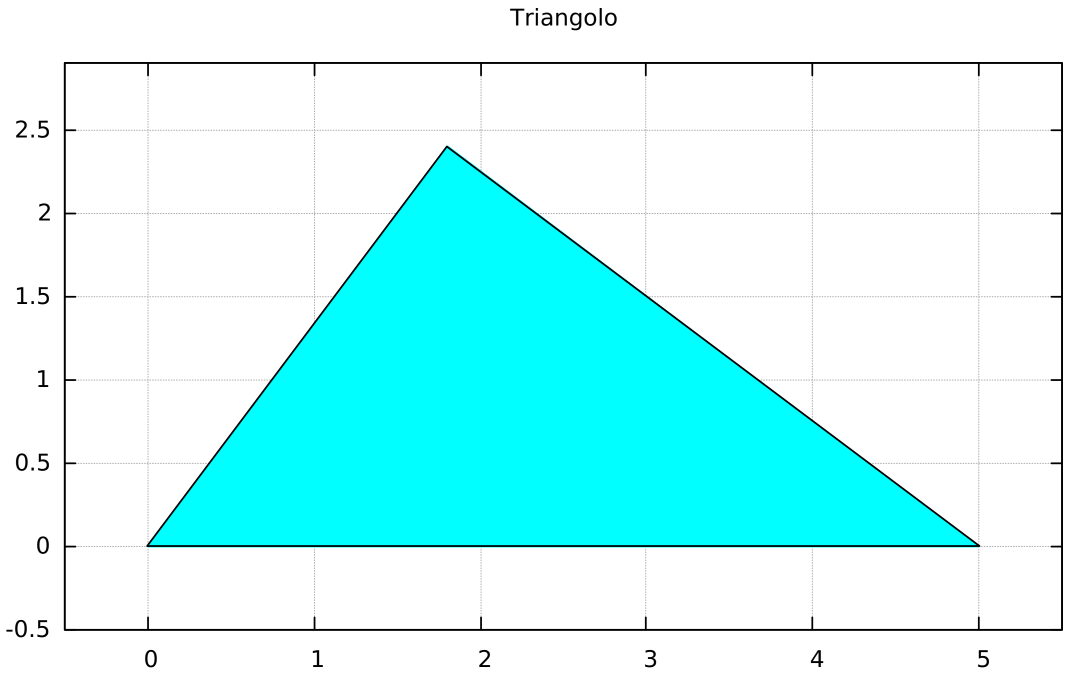


Figura 2: Triangolo.

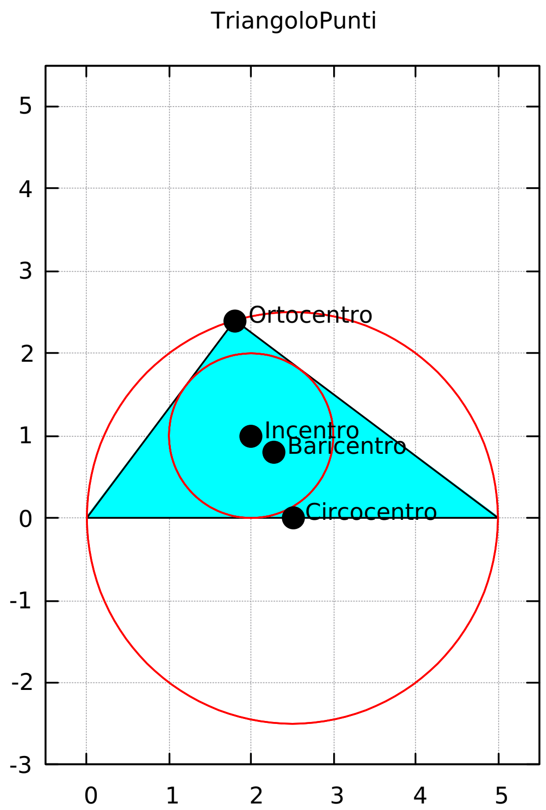


Figura 3: Triangolo con punti notevoli e le circonferenze inscritta e circoscritta.

Come si può notare il circocentro coincide col punto medio dell'ipotenusa, mentre l'ortocentro coincide col vertice corrispondete all'angolo retto, come previsto teoricamente.

Per testare la bontà delle funzioni che gestiscono i casi in cui vengono inseriti uno o due angoli, si è fatto eseguire il programma facendo in modo che esso ricostruisse un triangolo con una terna di lati 3,4,5 u.a. In tale modo si possono confrontare i risultati ottenuti con quelli che si ottengono inserendo direttamente i 3 lati.

Per il caso in cui l'utente deve inserire due lati e l'angolo compreso tra di essi, si è dato come input per i lati 3 e 4 u.a e per l'angolo 90°. Il programma ha costruito un triangolo con lati 3,4,5 u.a e con gli stessi angoli calcolati nel caso dei tre lati.

Per la parte in cui bisogna fornire due angoli ed un lato si è scelto di inserire 3 u.a come valore per il lato, mentre per gli angoli 90° e 53.13°. Nonostante il secondo angolo fosse un valore approssimato, il programma ha costruito un triangolo con lati 3,4,5 u.a, che dunque aveva le stesse caratteristiche del triangolo precedentemente costruito.

Appendici

A Formula di Erone

Siano p il semiperimetro del triangolo e a, b, c i suoi lati, allora l'area

$$A = \sqrt{p(p-a)(p-b)(p-c)} \tag{A.1}$$

B Teoremi utili

B.1 Teorema del seno

Siano a, b, c i lati del triangoli e α, β, γ i corrispondenti angoli opposti, allora vale:

$$\frac{a}{\sin\alpha} = \frac{b}{\sin\beta} = \frac{c}{\sin\gamma} \tag{B.1}$$

B.2 Teorema del coseno

Siano a, b, c i lati del triangoli e α, β, γ i corrispondenti angoli opposti, allora vale:

$$\begin{aligned} a^2 &= b^2 + c^2 - 2bc\cos\alpha \\ b^2 &= c^2 + a^2 - 2ca\cos\beta \\ c^2 &= a^2 + b^2 - 2ab\cos\gamma \end{aligned} \tag{B.2}$$

C Punti notevoli

C.1 Coordinate baricentro

Siano A=(x_A,y_A), B=(x_B,y_B), C=(x_C,y_C) i vertici e le loro rispettive coordinate, le coordinate del baricentro:

$$\begin{aligned} x_{Bari} &= \frac{x_A + x_B + x_C}{3} \\ y_{Bari} &= \frac{y_A + y_B + y_C}{3} \end{aligned} \tag{C.1}$$

C.2 Coordinate incentro e raggio circonferenza inscritta

Siano A=(x_A,y_A), B=(x_B,y_B), C=(x_C,y_C) i vertici e le loro rispettive coordinate, a, b, c le lunghezze dei lati, P il perimetro, A l'area, le coordinate dell'incentro

$$\begin{aligned} x_{In} &= \frac{a \cdot x_A + b \cdot x_B + c \cdot x_C}{P} \\ y_{In} &= \frac{a \cdot y_A + b \cdot y_B + c \cdot y_C}{P} \end{aligned} \tag{C.2}$$

il raggio della circonferenza inscritta:

$$R_{In} = 2 \cdot \frac{A}{P} \tag{C.3}$$

C.3 Coordinate circocentro e raggio circonferenza circoscritta

Siano A=(x_A,y_A), B=(x_B,y_B), C=(x_C,y_C) i vertici e le loro rispettive coordinate, a, b, c le lunghezze dei lati, A l'area, sia:

$$D = 2 \cdot [x_A(y_B - y_C) + x_B(y_C - y_A) + x_C(y_A - y_B)] \tag{C.4}$$

le coordinate del circocentro:

$$\begin{aligned} x_{Circo} &= \frac{[(x_A^2 + y_A^2)(y_B - y_C) + (x_B^2 + y_B^2)(y_C - y_A) + (x_C^2 + y_C^2)(y_A - y_B)]}{D} \\ y_{Circo} &= \frac{[(x_A^2 + y_A^2)(x_C - x_B) + (x_B^2 + y_B^2)(x_A - x_C) + (x_C^2 + y_C^2)(x_B - x_A)]}{D} \end{aligned} \tag{C.5}$$

il raggio della circonferenza circoscritta:

$$R_{Circo} = \frac{a \cdot b \cdot c}{4A} \tag{C.6}$$

C.4 Coordinate ortocentro

Siano A=(x_A,y_A), B=(x_B,y_B), C=(x_C,y_C) i vertici e le loro rispettive coordinate, le coordinate dell'ortocentro:

$$\begin{aligned} x_{Orto} &= \frac{[x_B(x_A - x_C) + y_B(y_A - y_C)][y_C - y_B] - [y_C - y_A][x_A(x_B - x_C) + y_A(y_B - y_C)]}{(x_C - x_B)(y_C - y_A) - (y_C - y_B)(x_C - x_A)} \\ y_{Orto} &= \frac{[x_B(x_A - x_C) + y_B(y_A - y_C)][x_C - x_B] - [x_C - x_A][x_A(x_B - x_C) + y_A(y_B - y_C)]}{(y_C - y_B)(x_C - x_A) - (x_C - x_B)(y_C - y_A)} \end{aligned} \tag{C.7}$$

D Calcolo altezze in un triangolo rettangolo con lati 3,4,5 u.a

Si consideri un triangolo rettangolo con lati di $a = 3$, $b = 4$ e $c = 5$ in unità arbitrarie. Siccome il triangolo è rettangolo le altezze relative ai lati a e b sono rispettivamente b ed a. Per trovare l'altezza relativa all'ipotenusa c, occorre sfruttare il secondo teorema di Euclide e risolvere il seguente sistema, dove m e n sono rispettivamente le proiezioni di a e b su c ed h è l'altezza relativa a c:

$$\begin{cases} m + n = c, \\ m^2 + h^2 = a^2, \\ n^2 + h^2 = b^2. \end{cases} \tag{D.1}$$

Ricavando m dalla prima e sostituendola nella seconda si ottiene $c^2 + n^2 + h^2 - 2cn = a^2$. Sottraendo la terza a quest'ultima si ottiene il valore di $n = (c^2 + b^2 - a^2)/2c = b^2/c$. Dall'ultima equazione del sistema si ha infine

$$h = b\sqrt{1 - \frac{b^2}{c^2}} \tag{D.2}$$

Sostituendo i valori $a = 3$, $b = 5$, $c = 5$ si ha

$$h = \frac{12}{5} = 2.4 \tag{D.3}$$