**Relazione di Laboratorio**

Corso: Sperimentazioni di Fisica 1 – LT

Anno accademico: 2013 – 2014

Docente: dott. C. Sada

Gruppo di Lavoro: Riccardo Milocco – 1070552 – riccardo.milocco@studenti.unipd.it

Andrea de Marco – 1069510 – andrea.demarco.2@studenti.unipd.it

Data di consegna della relazione: martedì, 1 aprile 2014

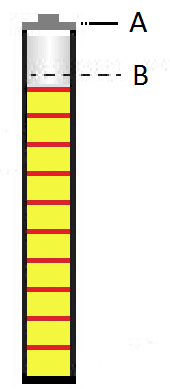
**Il VISCOSIMETRO**

**Obbiettivi dell’esperienza**:

L’esperienza ha lo scopo di calcolare il coefficiente di attrito viscoso di un determinato liquido fornito dal laboratorio: nel caso specifico, un liquido saponoso.

(R.M.)

**Apparato strumentale:**

****

- *Viscosimetro:* corpo cilindrico dotato di 10 tacche rosse orizzontali equidistanti l’una dall’altra a intervalli di 5 centimetri, e di un magnete rimovibile sull’ estremità superiore. Inoltre è stata applicata in una parte laterale del viscosimetro un foglio di carta gialla, per aumentare il contrasto tra le bande rosse e lo sfondo;

-*Magnete (A):*  comune corpo magnetizzato in grado di generare un campo magnetico. In aggiunta il nostro dispositivo era fornito di un imbuto esterno di plastica nel quale era inseribile una sfera in acciaio. Era provvisto di un pulsante laterale che, annullando il campo magnetico, lasciava cadere la pallina nell’apparato;

-*Sfere in acciaio:* 10 tipi di sfere, differenti in diametro: 1,5 mm, 2’’/32, 2,0 mm, 3’’/32, 4’’/32, 5’’/32, 6’’/32, 7’’/32, 8’’/32, 9’’/32, di cui sono stati utilizzati 7 tipi differenti, ovvero: 1,5 mm, 3’’/32, 5’’/32, 6’’/32, 7’’/32, 8’’/32, 9’’/32.

-*Cronometro manuale:* dispositivo digitale che ha permesso l’acquisizione dei tempi di caduta delle diverse sfere metalliche.  
In particolare, questo strumento era dotato della funzione “store” che permetteva di acquisire i tempi parziali, continuando comunque a registrare il tempo effettivo.

(R.M.)

**Metodologia di Misura**

Abbiamo deciso di utilizzare sette sfere di diverso diametro e registrarne i tempi di caduta in spazi prefissati in base ai nostri tempi di reazione, per evitare errori sistematici. In particolare le misurazioni sono state effettuate a partire dalle sfere di raggio minore poiché entrando nel fluido, le sfere introducono anche una piccola quantità d’aria: mentre le sfere con diametro maggiore riescono a rompere o deviare le bolle d’aria formatesi, le sfere di diametro minore tendono invece a “circumnavigarle”, cambiando notevolmente il moto. Pertanto le sfere di diametro minore sono state inserite per prime, non essendosi ancora formata la miscela aria-liquido saponoso.

In particolare, per le sfere di:

- 1,5 mm e 3’’/32, per ogni intervallo progressivo (5, 10, …, 50 cm) sono stati registrati i tempi di percorrenza, facendo partire il cronometro al passaggio della prima tacca a partire dall’estremità superiore, evitando errori di parallasse;

- 5’’/32 e 6’’/32, facendo partire il cronometro alla prima tacca dall’estremità superiore, sono stati registrati i tempi di percorrenza di ogni progressivo di 10 cm (10, 20, …, 50 cm);

- 7’’/32, facendo partire il cronometro alla seconda tacca dall’estremità superiore, sono stati rilevati i tempi di percorrenza di ogni progressivo di 15 cm (20, 35, 50 cm);

-8’’/32, sono stati misurati i due intervalli di tempi intercorsi tra la prima tacca e quella posta a 25 cm, e tra la prima tacca e l’ultima tacca, posta a 50cm.

-9’’/32, è stato misurato il tempo trascorso nel percorre lo spazio complessivo di 50 cm, facendo partire il cronometro, come nei casi precedenti, al passaggio della sfera in corrispondenza della prima tacca dall’alto.

E’ stato scelto di procedere con l’acquisizione dei campioni per ogni tipo di sfera, in base al tempo a disposizione e alla quantità di dati che avevamo previsto di raccogliere.

Dunque per le sfere di 1,5 mm e 3’’/32 sono stati raccolti 3 campioni, per le sfere di diametri superiori abbiamo cercato di raccoglierne il più possibile tenendo conto che alcune misurazioni sono andate perse per errori commessi. Pertanto sono stati raccolti 5 campioni per le sfere di diametro 5’’/32, 4 per le sfere di diametro 7’’/32, e 6 campioni per le altre sfere.

Infine ci è stato chiesto di registrare i tempi impiegati a percorre l’intera distanza di 50 cm da una sfera di diametro di 9’’/32 30 minuti dopo l’ultima rilevazione, per vedere come varia il tempo di percorrenza una volta che il sistema è tornato nelle condizioni iniziali (ovvero in assenza di bolle d’aria). Sono state utilizzate 5 sfere.

(A.d.M)

**Metodologia di Rielaborazione Dati**

Prima di tutto si sono determinati gli errori connessi agli spazi percorsi dalla sfera e ai tempi registrati.  
Per quanto riguarda l’errore dello spazio, abbiamo utilizzato la distribuzione triangolare ritenendo che il nostro occhio potesse registrare il passaggio della sfera a al disopra e al di sotto del traguardo fissato sullo strumento. Così a naso?

In seguito, abbiamo considerato come errore dei singoli tempi la sensibilità dello strumento pari a .

Fatto ciò è stata calcolata la velocità media della sfera, associando a ciascuna il proprio errore mediante la propagazione degli errori.

Quindi si è determinato il coefficiente d’attrito viscoso , che dall’analisi delle forze che agiscono sulla sfera in caduta risulta essere:

= (1.0)

Abbiamo preferito sfruttare la relazione tra diametro e raggio per evitare in seguito ulteriori calcoli, sapendo noto l’errore sul diametro. Quindi è stato calcolato l’errore commesso sulla stima di mediante la formula di propagazione degli errori

Successivamente per ottenere un’unica stima del coefficiente d’attrito viscoso che tenesse conto anche degli errori commessi, è stata calcolata la media pesata su tutti i risultati ottenuti e se ne è computato il relativo errore.

In secondo luogo, si è passati all’elaborazione di grafici che permettessero di osservare le dipendenze ottenute.   
Per verificare l’uniformità del moto, si è inserita in un grafico la dipendenza spazio-tempo: in ascissa sono stati riportati i tempi medi di percorrenza di un determinato intervallo, poiché presentano un errore minore rispetto agli spazi percorsi, posti in ordinata. Dal coefficiente angolare della retta interpolante i dati, si è trovata la velocità limite con la quale si sono potute confrontare le velocità medie di ogni intervallo, tramite compatibilità.

Inoltre, la velocità limite così calcolata ha permesso un’ulteriore stima del coefficiente d’attrito viscoso , ricavandolo direttamente dalla formula (1.0). Infine questa nuova stima è stata usata per verificare che le stime di fossero corrette e per valutare quale fosse il miglior metodo di stima del coefficiente di attrito viscoso.

(R.M.)

**Risultati sperimentali**

**Velocità medie per ogni intervallo**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 1,5mm | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  |  |  |  | NR |  |  |  |  |  |  |
|  |  |  |  | NR |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 3/32’’ | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  |  |  |  |  |  |  | NR |  |  | NR |
|  |  |  |  |  |  |  | NR |  |  | NR |
|  | NR |  |  |  |  |  |  |  |  |  |
|  | NR |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 5/32’’ | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 6/32’’ | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 7/32” | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | saltato | partenza |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | 0.0355 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 8/32’’ | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | Dati non registrati | | | |  | **Dati non registrati** | | | |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | 0.6 |
|  |  | | | |  |  | | | |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 9/32” | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | Dati non registrati | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  | | | | | | | | |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 9/32” con misurazioni dopo 30min | | | | | | | | | |
|  | Velocità medie parziali (m/s) e errori (m/s) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | Dati non registrati | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  | | | | | | | | |  |

**Coefficienti di attrito viscoso per ogni intervallo**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 1.5mm | | | | | | | | | |
|  | Coefficienti d’attrito parziali (N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | 4.1 | 4.1 | 4.2 | NR | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 |
|  | 0.3 |  |  | NR |  |  |  |  |  |  |
|  | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | NR |
|  |  |  |  |  |  |  |  |  |  | NR |
|  | 4.1 | 4.1 | 4.1 | 4.1 | 4.1 | 4.1 | 4.1 | 4.1 | 4.1 | 4.1 |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 3/32’’ | | | | | | | | | |
|  | Coefficienti d’attrito parziali (N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | 4.2 | NR | 4.2 | 4.2 | NR |
|  |  |  |  |  |  |  | NR |  |  | NR |
|  | NR | 4.1 | 4.2 | 4.2 | 4.2 | 4.2 | 4.3 | 4.2 | 4.3 | 4.3 |
|  | NR |  |  |  |  |  |  |  |  |  |
|  | 4.1 | 4.1 | 4.1 | 4.1 | 4.1 | 4.2 | 4.1 | 4.1 | 4.1 | 4.1 |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 5/32’’ | | | | | | | | | |
|  | Coefficienti d’attrito parziali (N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  |  | 3.9 |  | 3.9 |  | 3.9 |  | 3.9 |  | 3.9 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3.8 |  | 3.8 |  | 3.9 |  | 3.9 |  | 3.9 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3.9 |  | 5 |  | 4 |  | 4 |  | 4 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3.9 |  | 3.9 |  | 3.9 |  | 3.9 |  | 3.9 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3.8 |  | 3.9 |  | 3.9 |  | 3.9 |  | 3.9 |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 6/32’’ | | | | | | | | | |
|  | Coefficienti d’attrito parziali (N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  |  | 3.7 |  | 3.6 |  | 3.6 |  | 3.7 |  | 3.7 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3.5 |  | 3.6 |  | 3.6 |  | 3.6 |  | 3.7 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3.5 |  | 3.6 |  | 3.6 |  | 3.7 |  | NR |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  | 3.5 |  | 3.5 |  | 3.6 |  | 3.6 |  | 3.6 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | 3.5 |  | 3.6 |  | 3.6 |  | 3.6 |  | NR |
|  |  |  |  |  |  |  |  |  |  | NR |
|  |  | 3.5 |  | 3.5 |  | 3.6 |  | 3.6 |  | 3.6 |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 7/32” | | | | | | | | | |
|  | Coefficienti d’attrito parziali (N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | saltato | partenza |  | 3.4 |  |  | 3.4 |  |  | 3.4 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | 3.4 |  |  | 3.3 |  |  | 3.3 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | 3.3 |  |  | 3.3 |  |  | 3.3 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | 4 |  |  | 3.4 |  |  | 3.3 |
|  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 8/32’’ | | | | | | | | | |
|  | Coefficienti d’attrito parziali(N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | **Dati non registrati** | | | | 2.4 | **Dati non registrati** | | | | 2.6 |
|  |  |  |
|  | 2.5 | 2.7 |
|  |  |  |
|  | 2.4 | 2.7 |
|  |  |  |
|  | 2.7 | 2.7 |
|  |  |  |
|  | 2.4 | 2.7 |
|  |  |  |
|  | 2.7 | 2.6 |
|  |  | | | | 0.0020 |  | | | | 0.0018 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 9/32” | | | | | | | | | |
|  | Coefficienti d’attrito parziali (N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | **Dati non registrati** | | | | | | | | | 1.9 |
|  |  |
|  | 1.9 |
|  |  |
|  | 1.9 |
|  |  |
|  | 1.8 |
|  |  |
|  | 1.9 |
|  |  |
|  | 1.9 |
|  |  | | | | | | | | |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Diametro 9/32” con misurazioni dopo 30min | | | | | | | | | |
|  | Coefficienti d’attrito parziali (N\*s / m^2) e errori (N\*s / m^2) | | | | | | | | | |
|  | **1** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | **Dati non registrati** | | | | | | | | | 1.8 |
|  |  |
|  | 1.8 |
|  |  |
|  | 1.8 |
|  |  |
|  | 1.9 |
|  |  |
|  | 1.8 |
|  |  | | | | | | | | |  |

|  |  |  |
| --- | --- | --- |
| *Raggio* | *v’(m/s)* | *(m/s)* |
| 1.5 mm | 1.6\*10^-03 | 1\*10^-04 |
| 3/32” | 1.5\*10^-03 | 1.2\*10^-03 |
| 5/32” | 1.51\*10^-02 | 1\*10^-04 |
| 6/32” | 2.34\*10^-02 | 4\*10^-04 |
| 7/32” | 3.51\*10^-02 | 5\*10^-04 |
| 8/32” | NC | NC |
| 9/32” | NC | NC |
| 9/32” 30min | NC | NC |

|  |  |  |
| --- | --- | --- |
| *Raggio* | *’(N\*s/m)* | *(N\*s/m)* |
| 1.5 mm | 5.14\*10^+00 | 2.40\*10^-03 |
| 3/32” | 1.42\*10^+01 | 1.12\*10^+00 |
| 5/32” | 3.95\*10^+00 | 3.00\*10^-03 |
| 6/32” | 3.67\*10^+00 | 4.30\*10^-03 |
| 7/32” | 3.32\*10^+00 | 3.80\*10^-03 |
| 8/32” | 4.34\*10^+00 | 4.60\*10^-03 |
| 9/32” | NC | NC |
| 9/32” 30min | NC | NC |

|  |  |  |
| --- | --- | --- |
| *Raggio* | *da media pesata(N\*s/m)* | *da media pesata(N\*s/m)* |
| 1.5 mm | 4.14\*10^+00 | 4.00\*10^-04 |
| 3/32” | 4.17\*10^+00 | 4.00\*10^-04 |
| 5/32” | 3.92\*10^+00 | 5.00\*10^-04 |
| 6/32” | 3.5810^+00 | 4.00\*10^-04 |
| 7/32” | 3.34\*10^+00 | 6.00\*10^-04 |
| 8/32” | 2.54\*10^+00 | 6.00\*10^-04 |
| 9/32” | 1.86\*10^+00 | 6.00\*10^-04 |
| 9/32” 30min | 1.80\*10^+00 | 7.00\*10^-04 |

|  |  |  |
| --- | --- | --- |
| *Raggio* |  |  |
| 1.5 mm | 411.56 | No compatibilità |
| 3/32” | 8.9461 | No compatibilità |
| 5/32” | 7.9007 | No compatibilità |
| 6/32” | 18.6894 | No compatibilità |
| 7/32” | 4.89254 | No compatibilità |
| 8/32” | 389.156 | No compatibilità |
| 9/32” | NC | NC |
| 9/32” 30min | NC | NC |

|  |  |  |
| --- | --- | --- |
|  | *Raggio 1.5mm* |  |
| **velocità media(m/s)** |  | Compatibilità |
| 2\* | 0.4\* | 0.0027 - O |
| 2\* | 0.3\* | 0.0037 - O |
| 2\* | 0.3\* | 0.0045 - O |
| 2\* | 0.2\* | 0.0051 - O |
| 2\* | 0.2\* | 0.0056 - O |
| 2\* | 0.1\* | 0.0062 - O |
| 2\* | 0.1\* | 0.0067 - O |
| 2\* | 0.1\* | 0.0071 - O |
| 2\* | 0.1\* | 0.0075 - O |
| 2\* | 0.2\* | 0.0086 - O |

|  |  |  |
| --- | --- | --- |
|  | *Raggio 3/32”* |  |
| **velocità media(m/s)** |  | Compatibilità |
| 5.2\*10^-3 | 0.1 | 0.0524 - O |
| 5.2\*10^-3 | 0.7\* | 0.0742 - O |
| 5.2\*10^-3 | 0.4\* | 0.0903 - O |
| 5.2\*10^-3 | 0.3\* | 0.1039 - O |
| 5.1\*10^-3 | 0.3\* | 0.1157 - O |
| 5.1\*10^-3 | 0.2\* | 0.1262 - O |
| 5.1\*10^-3 | 0.2\* | 0.0515 - O |
| 5.1\*10^-3 | 0.2\* | 0.0551 - O |
| 5.1\*10^-3 | 0.2\* | 0.0584 - O |
| 5.1\*10^-3 | 0.1\* | 0.0467 - O |

|  |  |  |
| --- | --- | --- |
|  | Raggio 5/32” |  |
| **velocità media(m/s)** |  | Compatibilità |
| 1\* | 0.6 | 0.0005 - O |
| 2\* | 0.3 | 0.0004 - O |
| 2\* | 0.2 | 0.0003 - O |
| 2\* | 0.2 | 0.0003 - O |
| 2\* | 0.1 | 0.0002 - O |

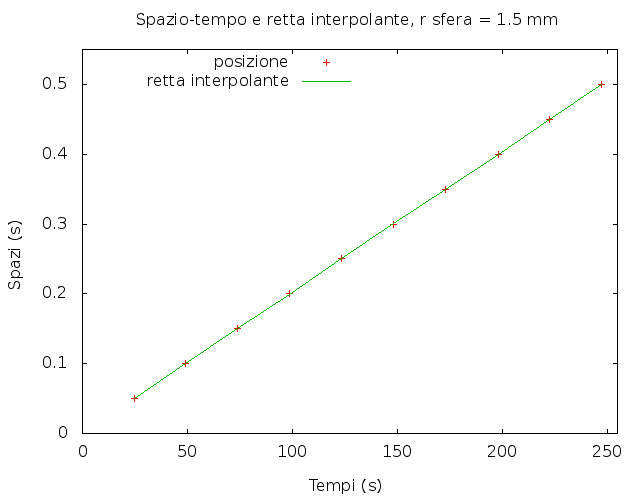
|  |  |  |
| --- | --- | --- |
|  | Raggio 6/32” |  |
| **velocità media(m/s)** |  | Compatibilità |
| 2\* | 1.6 | 0.0008 - O |
| 2\* | 0.8 | 0.0009 - O |
| 2\* | 0.5 | 0.0008 - O |
| 2\* | 0.4 | 0.0006 - O |
| 2\* | 0.3 | 0.0006 - O |

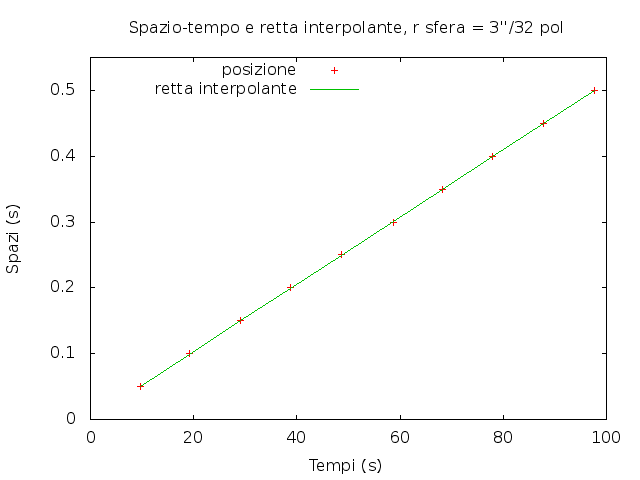
|  |  |  |
| --- | --- | --- |
|  | Raggio 7/32” |  |
| **velocità media(m/s)** |  | Compatibilità |
| 3.5 | 1.93 | 0.0003 - O |
| 3.5 | 0.978 | 0.0001 - O |
| 3.5 | 0.651 | 0.0002 - O |

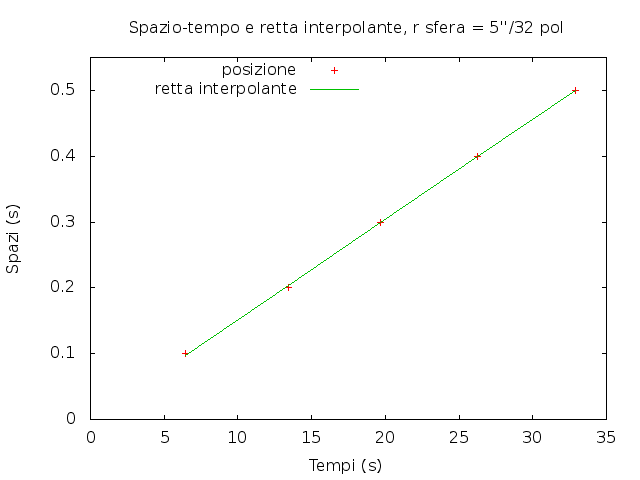
|  |  |  |
| --- | --- | --- |
| Raggio 8/32” | | |
| **velocità media(m/s)** |  |  |
| 0.0629 |  |  |
| 0.0577 |  |  |

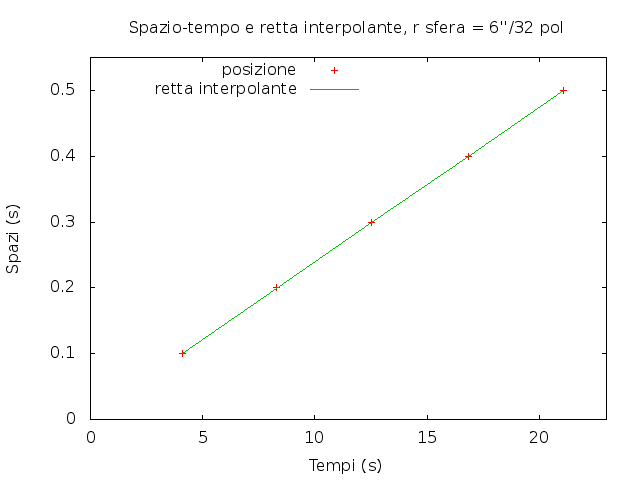
|  |  |  |
| --- | --- | --- |
| Raggio 9/32” | | |
| **velocità media(m/s)** |  |  |
| 0.0002 |  |  |

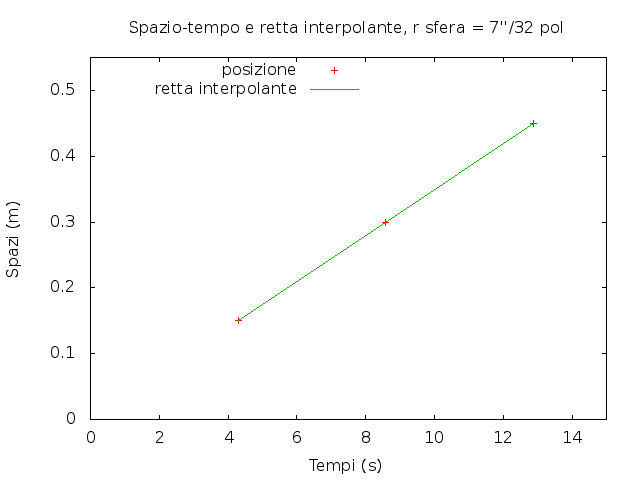
|  |  |  |
| --- | --- | --- |
| Raggio 9/32” dopo 30min | | |
| **velocità media(m/s)** |  |  |
| 0.0002 |  |  |











(R.M. – Grafici: A.d.M)

**Analisi statistica dei dati presentati**

Dai grafici ottenuti, si può osservare come la velocità del corpo in caduta nel liquido saponoso sia uniforme. Infatti, possiamo osservare che i singoli dati spazio-tempo si dispongono su una retta, il cui coefficiente angolare rappresenta la velocità limite.

Per quanto riguarda gli errori sistematici commessi, si ritiene che abbia influito sul moto di caduta l’aria trascinata dalle sfere all’interno del liquido saponoso: nel moto discendente le sfere possono incontrare delle bolle lasciate da quelle che le hanno precedute, facendo variare il coefficiente d’attrito viscoso computato.

In più, per verificare la linearità dei rapporti spazio-tempo si è determinata la compatibilità tra la velocità media di ogni intervallo e la velocità limite data dalla retta interpolante, con risultati sempre ottimali.

Tuttavia, non è stato sempre possibile avere un riscontro grafico dell’andamento della velocità. Per quanto riguarda le sfere di 9/32”, non si è riusciti a trovare una retta interpolante, in quanto il computo dei parametri dell’intercetta e del coefficiente angolare risultavano impossibili per legge matematica: è noto che per un solo punto passano infinite rette e, quindi avendo una sola stima della dipendenza spazio-tempo, non siamo in grado di determinare univocamente la retta interpolante.

Per quanto riguarda le sfere con raggio pari a 8/32”, la retta risultante non avrebbe portato a nessuna conclusione dal punto di vista statistico, dato il numero esiguo di dati disponibili.

Successivamente ottenute le due velocità, si sono computati i relativi coefficienti d’attrito viscoso. In questo caso, però, gli sono risultati statisticamente incompatibili anche se avessero valori approssimativamente simili: essendo gli errori infinitesimi di conseguenza anche la compatibilità dovrà risultare elevata.

Inoltre, è importante sottolineare che siccome per le sfere di raggio pari a 9/32” non si è riusciti a stimare la velocità limite dalla retta interpolante, non è possibile determinare un valore ideale del coefficiente d’attrito viscoso.

A conferma delle nostre ipotesi possiamo notare come la miscela aria-fluido saponoso abbia influito sul moto di caduta delle sfere osservando i risultati ottenuti lasciando cadere 5 sfere dopo 30 minuti dall’ultima misurazione effettuata notiamo che i valori ottenuti sono mediamente differenti:

(R.M.)

**Conclusioni**

Per quanto osservato finora, il miglior metodo di stima del coefficiente d’attrito, in assenza di significativi errori sistematici, è quello con il calcolo diretto della velocità dai dati, in quanto anche per le ultime due sfere è stato possibile computare il coefficiente d’attrito.

Tuttavia, se si analizza il solo caso della velocità limite si può riscontrare che la miglior stima della velocità si è ottenuta mediante l’interpolazione lineare dei dati misurati con una retta.  
Pertanto, non è propriamente corretto scartare un metodo a discapito dell’altro, poiché essi sono complementari e impiegati singolarmente possono fornire una valida metodologia di stima per la velocità limite e per il coefficiente d’attrito viscoso.

(R.M.)

**Appendici**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 1.5mm | | | | | | | | | |
| Tempi parziali (s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 24.5748 | 49.2909 | 74.3538 | NR | 124.2594 | 149.3293 | 174.2028 | 199.1658 | 224.1022 | 249.0596 |
| 24.7632 | 49.5107 | 74.3041 | 99.3342 | 124.1092 | 148.8978 | 173.6801 | 198.8262 | 223.5825 | NR |
| 24.3350 | 48.7988 | 73.4951 | 97.8454 | 122.5183 | 147.0253 | 171.7210 | 196.2040 | 220.7499 | 245.4327 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 3/32’’ | | | | | | | | | |
| Tempi parziali (s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 9.7746 | 19.4246 | 29.1285 | 38.8730 | 48.6177 | 58.3274 | NR | 77.9421 | 87.5620 | NR |
| NR | 19.0850 | 29.5624 | 39.5483 | 49.4027 | 59.3187 | 69.2351 | 79.0350 | 89.0258 | 98.9712 |
| 9.4770 | 19.0294 | 28.6075 | 38.0716 | 47.9479 | 58.6823 | 67.2264 | 76.9386 | 86.5118 | 96.4231 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 5/32’’ | | | | | | | | | |
| Tempi parziali (s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| NR | 6.5425 |  | 13.1489 |  | 19.7633 |  | 26.3778 |  | 33.0306 |
|  | 6.3051 |  | 12.8496 |  | 19.6053 |  | 26.1397 |  | 32.8168 |
|  | 6.5875 |  | 15.2103 |  | 19.8714 |  | 26.4561 |  | 33.1400 |
|  | 6.4914 |  | 13.0142 |  | 19.5675 |  | 26.1870 |  | 32.8803 |
|  | 6.4177 |  | 12.9079 |  | 19.6757 |  | 26.1713 |  | 32.7663 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 6/32’’ | | | | | | | | | |
| Tempi parziali (s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | 4.2648 |  | 8.4825 |  | 12.7206 |  | 17.0123 |  | 21.3575 |
|  | 4.0593 |  | 8.3277 |  | 12.5681 |  | 16.9109 |  | 21.1796 |
|  | 4.1329 |  | 8.3264 |  | 12.5997 |  | 17.0096 |  | NR |
|  | 4.0418 |  | 8.1812 |  | 12.4330 |  | 16.6849 |  | 20.8907 |
|  | 4.0788 |  | 8.2560 |  | 12.4817 |  | 16.8353 |  | NR |
|  | 4.0403 |  | 8.1787 |  | 12.4194 |  | 16.5962 |  | 20.9573 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 7/32” | | | | | | | | | |
| Tempi parziali (s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| saltato | partenza |  | 4.3670 |  |  | 8.6115 |  |  | 12.9046 |
|  |  |  | 4.3016 |  |  | 8.5077 |  |  | 12.8418 |
|  |  |  | 4.2201 |  |  | 8.5593 |  |  | 12.8397 |
|  |  |  | 4.3586 |  |  | 8.6031 |  |  | 12.8783 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 8/32’’ | | | | | | | | | |
| Tempi parziali(s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Dati non registrati | | | | 3.9051 | **Dati non registrati** | | | | 8.5361 |
| 4.0816 | 8.7715 |
| 3.9656 | 8.7067 |
| 3.9354 | 8.6791 |
| 4.0048 | 8.6717 |
| 3.9492 | 8.6136 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 9/32” | | | | | | | | | |
| Tempi parziali (s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Dati non registrati | | | | | | | | | 4.9764 |
| 4.9308 |
| 5.1316 |
| 4.8400 |
| 4.9721 |
| 4.9506 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diametro 9/32” con misurazioni dopo 30min | | | | | | | | | |
| Tempi parziali (s) +- | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Dati non registrati | | | | | | | | | 4.7258 |
| 4.7232 |
| 4.7630 |
| 4.9616 |
| 4.7340 |
| 4,7258 |

Programmi in C++:

- “visco.cxx”

- “medpes.cxx”

- “rettainterp.cxx”

(A.d.M)

**\*\* Codice del programma “visco.cxx” \*\***

#include <iostream>

#include <fstream>

#include <cmath>

#include <string>

#include <iomanip>

int main()

{

using namespace std;

string yesorno;

cout << "\n\n \*\*\*\*\*\* Esperimento Viscosimetro \*\*\*\*\*\*" << endl << endl;

do

{

char replydiametro;

cout << "\nDi quale sfera vuoi analizzare i dati?" << endl

<< "\na) Diametro ( 1,5 +- 0,010 ) mm" << endl

<< "b) Diametro ( 2,4 +- 0,010 ) mm" << endl

<< "c) Diametro ( 4,0 +- 0,010 ) mm" << endl

<< "d) Diametro ( 4,8 +- 0,010 ) mm" << endl

<< "e) Diametro ( 5,6 +- 0,010 ) mm" << endl

<< "f) Diametro ( 6,4 +- 0,010 ) mm" << endl

<< "g) Diametro ( 7,1 +- 0,010 ) mm" << endl

<< "h) Diametro ( 7,1 +- 0,010 ) mm dopo 30 min" << endl << endl;

cin >> replydiametro;

int totalemisure, mispercamp;

double deltas, diametro;

ifstream write;

if( replydiametro == 'a' )

{

write.open( "d1\_5mm.txt" );

totalemisure = 30;

deltas = 0.05;

mispercamp = 10;

diametro = 0.0015;

}

else if( replydiametro == 'b' )

{

write.open( "d3\_32pol.txt" );

totalemisure = 30;

deltas = 0.05;

mispercamp = 10;

diametro = 0.0024;

}

else if( replydiametro == 'c' )

{

write.open( "d5\_32pol.txt" );

totalemisure = 25;

deltas = 0.1;

mispercamp = 5;

diametro = 0.0040;

}

else if( replydiametro == 'd' )

{

write.open( "d6\_32pol.txt" );

totalemisure = 30;

deltas = 0.1;

mispercamp = 5;

diametro = 0.0048;

}

else if( replydiametro == 'e' )

{

write.open( "d7\_32pol.txt" );

totalemisure = 12;

deltas = 0.15;

mispercamp = 3;

diametro = 0.0056;

}

else if( replydiametro == 'f' )

{

write.open( "d8\_32pol.txt" );

totalemisure = 12;

deltas = 0.25;

mispercamp = 2;

diametro = 0.0064;

}

else if( replydiametro == 'g' )

{

write.open( "d9\_32pol.txt" );

totalemisure = 6;

deltas = 0.5;

mispercamp = 1;

diametro = 0.0071;

}

else if( replydiametro == 'h' )

{

write.open( "d9\_32pol30min.txt" );

totalemisure = 5;

deltas = 0.5;

mispercamp = 1;

diametro = 0.0071;

}

double tempi[totalemisure];

for( int i = 0; i < totalemisure; i++ )

write >> tempi[i];

write.close;

double velocita[totalemisure];

int j = 0;

for( int i = 0; i < totalemisure; i++ )

{

if( tempi[i] == 0 )

velocita[i] = 0;

else if( tempi[i] != 0 )

velocita[i] = ( j + 1 ) \* deltas / tempi[i];

j++;

if( j >= mispercamp )

j = 0;

}

cout << "\nNPR = Non e' stato possibile registrare questo dato" << endl << endl;

int campioni = totalemisure / mispercamp;

cout << " Campione 1" << " ";

for( int l = 1; l < campioni; l++)

{

if( l < campioni - 1)

cout << " Campione " << l+1 << " ";

else if( l == campioni - 1 )

cout << " Campione " << l+1 << endl << endl << " ";

}

// SERVE SOLO PER STAMPARE

int k = 0; // k contatore definisce l'orizzontale

// i definisce la verticale

for( int i = 0; i < mispercamp; i++ )

{

for( int j = 0; j < campioni; j++ )

{

if( tempi[( i + k \* mispercamp )] == 0 )

{

if( 0 <= ( i + 1 + k \* mispercamp ) && ( i + 1 + k \* mispercamp ) <= 9 ) // serve per intabulare maniacalmente

cout << "v" << i + 1 + k \* mispercamp << ") NPR "; // i dati

else

cout << "\bv" << i + 1 + k \* mispercamp << ") NPR ";

}

else if( tempi[( i + k \* mispercamp )] != 0 )

{

if( 0 <= ( i + 1 + k \* mispercamp ) && ( i + 1 + k \* mispercamp ) <= 9 )

cout << setiosflags( ios::fixed | ios:: showpoint ) << setprecision(5)

<< "v" << i + 1 + k \* mispercamp << ") = " << velocita[( i + k \* mispercamp )] << " ";

else

cout << "\bv" << i + 1 + k \* mispercamp << ") = " << velocita[( i + k \* mispercamp )] << " ";

}

k++;

if( k == campioni )

{

cout << endl << " ";

k = 0;

}

}

}

double errorediametro = 0.000010;

double gravita = 9.806;

double erroregravita = 0.001;

double rhosf = 7870; // kg/m^3

double errorerhosf = 5; // kg/m^3

double rholiq = 1032; // kg/m^3

double errorerholiq = 1;// kg/m^3

double diffrho = rhosf - rholiq;

double eta[totalemisure];

for( int i = 0; i < totalemisure; i++ )

{

if( velocita[i] == 0 )

eta[i] = 0;

else if( velocita[i] != 0 )

eta[i] = pow( diametro, 2 ) \* gravita \* diffrho / ( 18 \* velocita[i] ) ;

}

cout << endl << endl << " ";

for( int i = 0; i < mispercamp; i++ )

{

for( int j = 0; j < campioni; j++ )

{

if( tempi[( i + k \* mispercamp )] == 0 )

{

if( 0 <= ( i + 1 + k \* mispercamp ) && ( i + 1 + k \* mispercamp ) <= 9 )

cout << "eta" << i + 1 + k \* mispercamp << ") NPR ";

else

cout << "\beta" << i + 1 + k \* mispercamp << ") NPR ";

}

else if( tempi[( i + k \* mispercamp )] != 0 )

{

if( 0 <= ( i + 1 + k \* mispercamp ) && ( i + 1 + k \* mispercamp ) <= 9 )

cout << setiosflags( ios::fixed | ios:: showpoint ) << setprecision(5)

<< "eta" << i + 1 + k \* mispercamp << ") = " << eta[( i + k \* mispercamp )] << " ";

else

cout << "\beta" << i + 1 + k \* mispercamp << ") = " << eta[( i + k \* mispercamp )] << " ";

}

k++;

if( k == campioni )

{

cout << endl << " ";

k = 0;

}

}

}

double erroredeltas = 0.002 / sqrt(6);

double sigmav[totalemisure];

double addendosv1[totalemisure];

double addendosv2[totalemisure];

double erroretempi = 0.0001;

for( int i = 0; i < totalemisure; i++ )

{

if( tempi[i] == 0 )

{

addendosv1[i] = 0;

addendosv2[i] = 0;

}

else if( tempi[i] != 0 )

{

addendosv1[i] = pow( erroredeltas, 2 ) / ( tempi[i] \* tempi[i] );

addendosv2[i] = pow( deltas, 2 ) \* pow( erroretempi, 2 ) / ( tempi[i] \* tempi[i] \* tempi[i] \* tempi[i] );

}

}

for( int i = 0; i < totalemisure; i++ )

sigmav[i] = sqrt( addendosv1[i] + addendosv2[i] );

ofstream output, read, fout;

if( replydiametro == 'a' )

{

read.open( "velocita1\_5mm.txt" );

fout.open( "sigmav1\_5mm.txt" );

output.open( "eta1\_5mm.txt" );

}

else if( replydiametro == 'b' )

{

read.open( "velocita3\_32pol.txt" );

fout.open( "sigmav3\_32pol.txt" );

output.open( "eta3\_32pol.txt" );

}

else if( replydiametro == 'c' )

{

read.open( "velocita5\_32pol.txt" );

fout.open( "sigmav5\_32pol.txt" );

output.open( "eta5\_32pol.txt" );

}

else if( replydiametro == 'd' )

{

read.open( "velocita6\_32pol.txt" );

fout.open( "sigmav6\_32pol.txt" );

output.open( "eta6\_32pol.txt" );

}

else if( replydiametro == 'e' )

{

read.open( "velocita7\_32pol.txt" );

fout.open( "sigmav7\_32pol.txt" );

output.open( "eta7\_32pol.txt" );

}

else if( replydiametro == 'f' )

{

read.open( "velocita8\_32pol.txt" );

fout.open( "sigmav8\_32pol.txt" );

output.open( "eta8\_32pol.txt" );

}

else if( replydiametro == 'g' )

{

read.open( "velocita9\_32pol.txt" );

fout.open( "sigmav9\_32pol.txt" );

output.open( "eta9\_32pol.txt" );

}

else if( replydiametro == 'h' )

{

read.open( "velocita9\_32pol30min.txt" );

fout.open( "sigmav9\_32pol30min.txt" );

output.open( "eta9\_32pol30min.txt" );

}

for( int i = 0; i < totalemisure; i++ )

{

read << velocita[i] << endl;

fout << sigmav[i] << endl;

output << eta[i] << endl;

}

cout << "\n\nVuoi rieseguire il programma? ";

cin >> yesorno;

do

{

if( yesorno != "si" && yesorno != "no" )

{

cout << "\n\*\*\* Risposta non valida \*\*\* \nVuoi rieseguire il programma? ";

cin >> yesorno;

}

} while( yesorno != "si" && yesorno != "no" );

}

while( yesorno == "si" );

return 0;

}

**\*\* Codice del programma “medpes.cxx” \*\***

#include <iostream>

#include <fstream>

#include <cmath>

#include <string>

#include <iomanip>

double sommatoria( double \*, double \*, int );

int main()

{

using namespace std;

string yesorno;

cout << "\n\n \*\*\*\*\*\* Esperimento Viscosimetro \*\*\*\*\*\*" << endl << endl;

do

{

char replydiametro;

cout << "\nDi quale sfera vuoi analizzare i dati?" << endl

<< "\na) Diametro ( 1,5 +- 0,010 ) mm" << endl

<< "b) Diametro ( 2,4 +- 0,010 ) mm" << endl

<< "c) Diametro ( 4,0 +- 0,010 ) mm" << endl

<< "d) Diametro ( 4,8 +- 0,010 ) mm" << endl

<< "e) Diametro ( 5,6 +- 0,010 ) mm" << endl

<< "f) Diametro ( 6,4 +- 0,010 ) mm" << endl

<< "g) Diametro ( 7,1 +- 0,010 ) mm" << endl

<< "h) Diametro ( 7,1 +- 0,010 ) mm dopo 30 min" << endl << endl;

cin >> replydiametro;

ifstream write, input, fin, gin, hin, tit;

int totalemisure;

double diametro;

if( replydiametro == 'a' )

{

fin.open( "d1\_5mm.txt" );

input.open( "velocita1\_5mm.txt" );

gin.open( "sigmav1\_5mm.txt" );

write.open( "eta1\_5mm.txt" );

hin.open( "b1\_5mm.txt" );

tit.open( "errb1\_5mm.txt" );

totalemisure = 30;

diametro = 0.0015;

}

else if( replydiametro == 'b' )

{

fin.open( "d3\_32pol.txt" );

input.open( "velocita3\_32pol.txt" );

gin.open( "sigmav3\_32pol.txt" );

write.open( "eta3\_32pol.txt" );

hin.open( "b3\_32pol.txt" );

tit.open( "errb3\_32pol.txt" );

totalemisure = 30;

diametro = 0.0024;

}

else if( replydiametro == 'c' )

{

fin.open( "d5\_32pol.txt" );

input.open( "velocita5\_32pol.txt" );

gin.open( "sigmav5\_32pol.txt" );

write.open( "eta5\_32pol.txt" );

hin.open( "b5\_32pol.txt" );

tit.open( "errb5\_32pol.txt" );

totalemisure = 25;

diametro = 0.0040;

}

else if( replydiametro == 'd' )

{

fin.open( "d6\_32pol.txt" );

input.open( "velocita6\_32pol.txt" );

gin.open( "sigmav6\_32pol.txt" );

write.open( "eta6\_32pol.txt" );

hin.open( "b6\_32pol.txt" );

tit.open( "errb6\_32pol.txt" );

totalemisure = 30;

diametro = 0.0048;

}

else if( replydiametro == 'e' )

{

fin.open( "d7\_32pol.txt" );

input.open( "velocita7\_32pol.txt" );

gin.open( "sigmav7\_32pol.txt" );

write.open( "eta7\_32pol.txt" );

hin.open( "b7\_32pol.txt" );

tit.open( "errb7\_32pol.txt" );

totalemisure = 12;

diametro = 0.0056;

}

else if( replydiametro == 'f' )

{

fin.open( "d8\_32pol.txt" );

input.open( "velocita8\_32pol.txt" );

gin.open( "sigmav8\_32pol.txt" );

write.open( "eta8\_32pol.txt" );

totalemisure = 12;

diametro = 0.0064;

}

else if( replydiametro == 'g' )

{

fin.open( "d9\_32pol.txt" );

input.open( "velocita9\_32pol.txt" );

gin.open( "sigmav9\_32pol.txt" );

write.open( "eta9\_32pol.txt" );

totalemisure = 6;

diametro = 0.0071;

}

else if( replydiametro == 'h' )

{

fin.open( "d9\_32pol30min.txt" );

input.open( "velocita9\_32pol30min.txt" );

gin.open( "sigmav9\_32pol30min.txt" );

write.open( "eta9\_32pol30min.txt" );

totalemisure = 5;

diametro = 0.0071;

}

double tempi[totalemisure];

double velocita[totalemisure];

double eta[totalemisure];

double sigmav[totalemisure];

double b, sigma\_b;

for( int i = 0; i < totalemisure; i++ )

{

fin >> tempi[i];

input >> velocita[i];

gin >> sigmav[i];

write >> eta[i];

}

hin >> b;

tit >> sigma\_b;

fin.close();

input.close();

gin.close();

write.close();

hin.close();

double g = 9.806;

double erroreg = 0.001;

double rho = 7870.0;

double errorerho = 5.0;

double rholiq = 1032.0;

double errorerholiq = 1;

double errorediametro = 0.000010;

double erroretaquad[totalemisure];

double add1[totalemisure], add2[totalemisure], add3[totalemisure], add4[totalemisure];

for( int i = 0; i < totalemisure; i++ )

{

if( velocita[i] != 0 )

{

add1[i] = 4 \* pow( g, 2 ) \* pow( ( rho - rholiq ), 2 ) \* pow( errorediametro, 2 );

add2[i] = pow( diametro, 2 ) \* pow( ( rho - rholiq ), 2 ) \* pow( erroreg, 2 );

add3[i] = pow( diametro, 2 ) \* pow( g, 2 ) \* pow( ( rho - rholiq ), 2 ) \* pow( sigmav[i], 2 ) / pow( velocita[i], 2 );

add4[i] = pow( diametro, 2 ) \* pow( g, 2 ) \* ( pow( errorerho, 2 ) + pow( errorerholiq, 2 ) );

erroretaquad[i] = pow( diametro, 2 ) \* sqrt( add1[i] + add2[i] + add3[i] + add4[i] ) / ( 324 \* velocita[i]);

}

else if( velocita[i] == 0 )

erroretaquad[i] = 0;

cout << endl << erroretaquad[i] << endl;

}

double etasuerretaquad[totalemisure];

double peso[totalemisure];

for( int i = 0; i < totalemisure; i++ )

{

if( erroretaquad[i] == 0 )

{

etasuerretaquad[i] = 0;

peso[i] = 0;

}

else if( erroretaquad[i] != 0 )

{

etasuerretaquad[i] = eta[i] / erroretaquad[i];

peso[i] = 1 / erroretaquad[i];

}

}

double mediapesataeta;

double somma1 = sommatoria( etasuerretaquad, etasuerretaquad + totalemisure, totalemisure );

double somma2 = sommatoria( peso, peso + totalemisure, totalemisure );

mediapesataeta = somma1 / somma2;

double erroremediapesataeta;

erroremediapesataeta = sqrt( 1 / somma2 );

cout << "\n\nMedia pesata degli eta = " << mediapesataeta << endl

<< "Errore della media pesata = " << erroremediapesataeta << endl;

//COMPATIBILITA'

double etadaretta = pow( diametro, 2 ) \* 9.806 \* ( rho - rholiq ) / ( 18 \* b );

double erroretadaretta = sqrt( pow( diametro, 2 ) \*

sqrt( 4 \* pow( g, 2 ) \* pow( ( rho - rholiq ), 2 ) \* pow( errorediametro, 2 ) +

pow( diametro, 2 ) \* pow( ( rho - rholiq ), 2 ) \* pow( erroreg, 2 ) +

pow( diametro, 2 ) \* pow( g, 2 ) \* pow( ( rho - rholiq ), 2 ) \* pow( sigma\_b, 2 ) / pow( b, 2 ) +

pow( diametro, 2 ) \* pow( g, 2 ) \* ( pow( errorerho, 2 ) + pow( errorerholiq, 2 ) ) )

/ ( 324 \* b) );

cout << "eta da retta: " << etadaretta << endl

<< "errore eta da retta: " << erroretadaretta << endl

<< "\nCompatibilita' eta" << endl << endl;

double compatibilita = 0;

compatibilita = fabs( mediapesataeta - etadaretta ) / sqrt( pow( erroremediapesataeta, 2 ) + pow( erroretadaretta, 2 ) );

cout << compatibilita << setw(37);

if( compatibilita < 1 ) cout << "OTTIMA!" << endl;

else if( 1 <= compatibilita && compatibilita < 2 ) cout << "BUONA" << endl;

else if( 2 <= compatibilita && compatibilita < 3 ) cout << "pessima" << endl;

else if( 3 < compatibilita ) cout << "Non c'e' compatibilita'" << endl;

cout << "\n\nVuoi rieseguire il programma? ";

cin >> yesorno;

do

{

if( yesorno != "si" && yesorno != "no" )

{

cout << "\n\*\*\* Risposta non valida \*\*\* \nVuoi rieseguire il programma? ";

cin >> yesorno;

}

} while( yesorno != "si" && yesorno != "no" );

}

while( yesorno == "si" );

return 0;

}

double sommatoria( double \*inizio, double \*fine, int totalemisure )

{

double sum = 0;

double \*ptr;

for( ptr = inizio; ptr != fine; ptr++ )

sum = sum + \*ptr;

return sum;

}

**\*\* Codice del programma “rettainterp.cxx” \*\***

#include <iostream>

#include <fstream>

#include <cmath>

#include <string>

#include <iomanip>

double sommatoria( double \*, double \*, int );

int main()

{

using namespace std;

string yesorno;

cout << "\n\n \*\*\*\*\*\* Esperimento Viscosimetro \*\*\*\*\*\*" << endl << endl;

do

{

char replydiametro;

cout << "\nDi quale sfera vuoi calcolare i valori della retta interpolante s = s0 + vt?" << endl

<< "\na) Diametro ( 1,5 +- 0,010 ) mm" << endl

<< "b) Diametro ( 2,4 +- 0,010 ) mm" << endl

<< "c) Diametro ( 4,0 +- 0,010 ) mm" << endl

<< "d) Diametro ( 4,8 +- 0,010 ) mm" << endl

<< "e) Diametro ( 5,6 +- 0,010 ) mm" << endl endl;

cin >> replydiametro;

int totalemisure, mispercamp;

double deltas, diametro;

ifstream write;

if( replydiametro == 'a' )

{

write.open( "d1\_5mm.txt" );

totalemisure = 30;

deltas = 0.05;

mispercamp = 10;

diametro = 0.0015;

}

else if( replydiametro == 'b' )

{

write.open( "d3\_32pol.txt" );

totalemisure = 30;

deltas = 0.05;

mispercamp = 10;

diametro = 0.0024;

}

else if( replydiametro == 'c' )

{

write.open( "d5\_32pol.txt" );

totalemisure = 25;

deltas = 0.1;

mispercamp = 5;

diametro = 0.0040;

}

else if( replydiametro == 'd' )

{

write.open( "d6\_32pol.txt" );

totalemisure = 30;

deltas = 0.1;

mispercamp = 5;

diametro = 0.0048;

}

else if( replydiametro == 'e' )

{

write.open( "d7\_32pol.txt" );

totalemisure = 12;

deltas = 0.15;

mispercamp = 3;

diametro = 0.0056;

}

double tempi[totalemisure];

for( int i = 0; i < totalemisure; i++ )

write >> tempi[i];

write.close;

double tempimedi[mispercamp];

int campioni = totalemisure / mispercamp;

int counter = 0;

cout << "\nTempi medi" << endl << endl;

for( int i = 0; i < mispercamp; i++ )

{

for( int k = 0; k < campioni; k++ )

{

tempimedi[i] += tempi[i + k \* mispercamp];

if( tempi[i + k \* mispercamp] == 0 )

counter++;

}

tempimedi[i] = tempimedi[i] / ( campioni - counter );

counter = 0;

cout << tempimedi[i] << endl;

}

double spazi[mispercamp];

for( int i = 0; i < mispercamp; i++ )

spazi[i] = deltas \* ( i + 1 );

cout << "\nVelocita' medie" << endl << endl;

double velocitamed[mispercamp];

for( int i = 0; i < mispercamp; i++ )

{

velocitamed[i] = spazi[i] / tempimedi[i];

cout << setiosflags( ios::fixed | ios::showpoint ) << setprecision(4) << velocitamed[i] << endl;

}

double spazitempi[mispercamp];

double tempiquadro[mispercamp];

for( int i = 0; i < mispercamp; i++ )

{

tempiquadro[i] = pow( tempi[i], 2 );

spazitempi[i] = tempimedi[i] \* spazi[i];

}

double delta, sommatempiquadro, sommatempi, sommaspazitempi, sommaspazi;

sommatempiquadro = sommatoria( tempiquadro, tempiquadro + mispercamp, mispercamp );

sommatempi = sommatoria( tempimedi, tempimedi + mispercamp, mispercamp );

delta = mispercamp \* sommatempiquadro - pow ( sommatempi, 2 );

sommaspazi = sommatoria( spazi, spazi + mispercamp, mispercamp );

sommaspazitempi = sommatoria( spazitempi, spazitempi + mispercamp, mispercamp );

double a, addendo\_1a, addendo\_2a;

addendo\_1a = sommatempiquadro \* sommaspazi;

addendo\_2a = sommatempi \* sommaspazitempi;

a = ( 1.0 / delta ) \* ( addendo\_1a - addendo\_2a );

double b, addendo\_1b, addendo\_2b;

addendo\_1b = mispercamp \* sommaspazitempi;

addendo\_2b = sommatempi \* sommaspazi;

b = ( 1.0 / delta ) \* ( addendo\_1b - addendo\_2b );

double scartiretta[mispercamp];

for ( int i = 0; i < mispercamp; i++ )

scartiretta[i] = pow ( ( a + b \* tempimedi[i] - spazi[i] ), 2 );

double sigma\_y, sigma\_a, sigma\_b;

sigma\_y = sqrt( sommatoria ( scartiretta, scartiretta + mispercamp, mispercamp ) / ( mispercamp - 2 ) );

sigma\_a = sigma\_y \* sqrt ( sommatempiquadro / delta );

sigma\_b = sigma\_y \* sqrt ( mispercamp / delta );

cout << "\n\na: " << a << endl

<< "sigma a: " << sigma\_a << endl

<< "v: " << b << endl

<< "sigma v: " << sigma\_b << endl

<< "sigma y: " << sigma\_y << endl << endl;

ofstream read, fout, gout;

if( replydiametro == 'a' )

{

read.open( "ri1\_5mm.txt" );

fout.open( "b1\_5mm.txt");

gout.open( "errb1\_5mm.txt");

}

else if( replydiametro == 'b' )

{

read.open( "ri3\_32pol.txt" );

fout.open( "b3\_32pol.txt");

gout.open( "errb3\_32pol.txt");

}

else if( replydiametro == 'c' )

{

read.open( "ri5\_32pol.txt" );

fout.open( "b5\_32pol.txt");

gout.open( "errb5\_32pol.txt");

}

else if( replydiametro == 'd' )

{

read.open( "ri6\_32pol.txt" );

fout.open( "b6\_32pol.txt");

gout.open( "errb6\_32pol.txt");

}

else if( replydiametro == 'e' )

{

read.open( "ri7\_32pol.txt" );

fout.open( "b7\_32pol.txt");

gout.open( "errb7\_32pol.txt");

}

else if( replydiametro == 'f' )

{

read.open( "ri8\_32pol.txt" );

fout.open( "b8\_32pol.txt");

gout.open( "errb8\_32pol.txt");

}

else if( replydiametro == 'g' )

{

read.open( "ri9\_32pol.txt" );

fout.open( "b9\_32pol.txt");

gout.open( "errb9\_32pol.txt");

}

else if( replydiametro == 'h' )

{

read.open( "ri9\_32pol30min.txt" );

fout.open( "b9\_32pol30min.txt");

gout.open( "errb9\_32pol30min.txt");

}

for( int i = 0; i < mispercamp; i++ )

read << tempimedi[i] << " " << a + b \* tempimedi[i] << endl;

fout << b << endl;

gout << sigma\_b << endl;

double erroretempi[mispercamp];

double sommascartiq;

for( int i = 0; i < mispercamp; i++ )

{

for( int k = 0; k < campioni; k++ )

sommascartiq += pow( ( tempi[i + k \* mispercamp] - tempimedi[i] ) , 2 );

erroretempi[i] = sqrt( sommascartiq / mispercamp \* ( mispercamp - 1 ) );

}

double sigmavm[mispercamp];

double errorespazi = 0.002 / sqrt(6);

for( int i = 0; i < mispercamp; i++ )

sigmavm[i] = sqrt( pow( ( errorespazi / tempimedi[i]), 2 ) + pow( spazi[i] \* erroretempi[i], 2 ) / pow( tempimedi[i], 4 ) );

//COMPATIBILITA'

cout << "Compatibilita' velocita'" << endl << endl;

for ( int i = 0; i < mispercamp; i++ )

{

double compatibilita = 0;

compatibilita = fabs( velocitamed[i] - b ) / sqrt( sigmavm[i] + pow( sigma\_b, 2 ) );

cout << compatibilita << setw(37);

if( compatibilita < 1 ) cout << "OTTIMA!" << endl;

else if( 1 <= compatibilita && compatibilita < 2 ) cout << "BUONA" << endl;

else if( 2 <= compatibilita && compatibilita < 3 ) cout << "pessima" << endl;

else if( 3 < compatibilita ) cout << "Non c'e' compatibilita'" << endl;

}

cout << "\n\nVuoi rieseguire il programma? ";

cin >> yesorno;

do

{

if( yesorno != "si" && yesorno != "no" )

{

cout << "\n\*\*\* Risposta non valida \*\*\* \nVuoi rieseguire il programma? ";

cin >> yesorno;

}

} while( yesorno != "si" && yesorno != "no" );

}

while( yesorno == "si" );

return 0;

}

double sommatoria( double \*inizio, double \*fine, int totalemisure )

{

double sum = 0;

double \*ptr;

for( ptr = inizio; ptr != fine; ptr++ )

sum = sum + \*ptr;

return sum;

}