Simulazione di uno sciame elettromagnetico

Progetto finale Metodi Computazionali per la Fisica

Lorenzo Spera [1]

Università degli Studi di Perugia [1]
Corso di Laurea Triennale in Fisica

Indice

Indice

Struttura del progetto

Struttura

Per realizzare la simulazione dello sciame elettromagnetico adattando opportunamente il modello di Rossi si sono realizzati due script. Uno che implementasse la simulazione e l'altro che permettesse di testarla.

Struttura del progetto

Struttura

Per realizzare la simulazione dello sciame elettromagnetico adattando opportunamente il modello di Rossi si sono realizzati due script. Uno che implementasse la simulazione e l'altro che permettesse di testarla.

- classi per la creazione e gestione delle particelle.

Struttura del progetto

Struttura

Per realizzare la simulazione dello sciame elettromagnetico adattando opportunamente il modello di Rossi si sono realizzati due script. Uno che implementasse la simulazione e l'altro che permettesse di testarla.

- classi per la creazione e gestione delle particelle.
- classe per la definizione di un'istanza sciame elettromagnetico.

Struttura del progetto

Struttura

Per realizzare la simulazione dello sciame elettromagnetico adattando opportunamente il modello di Rossi si sono realizzati due script. Uno che implementasse la simulazione e l'altro che permettesse di testarla.

- classi per la creazione e gestione delle particelle.
- classe per la definizione di un'istanza sciame elettromagnetico.
- simulazione dello sciame a partire da valori scelti dall'utente.

Struttura del progetto

Struttura

Per realizzare la simulazione dello sciame elettromagnetico adattando opportunamente il modello di Rossi si sono realizzati due script. Uno che implementasse la simulazione e l'altro che permettesse di testarla.

- classi per la creazione e gestione delle particelle.
- classe per la definizione di un'istanza sciame elettromagnetico.
- simulazione dello sciame a partire da valori scelti dall'utente.
- risultati della simulazione nello script `run_sciame.py`.

Struttura del progetto

Scelta dei parametri

I valori scelti dall'utente sono i seguenti: energia della particella iniziale E_0, energia critica del materiale E_c, perdita per ionizzazione dEx_0 e il passo di avanzamento $s \in [0, 1]$. La procedura è stata la seguente:

Struttura del progetto

Scelta dei parametri

I valori scelti dall'utente sono i seguenti: energia della particella iniziale E_0, energia critica del materiale E_c, perdita per ionizzazione dEx_0 e il passo di avanzamento $s \in [0, 1]$. La procedura è stata la seguente:

- si sono selezionati due materiali diversi: acqua liquida e silicato di bismuto.

Struttura del progetto

Scelta dei parametri

I valori scelti dall'utente sono i seguenti: energia della particella iniziale E_0, energia critica del materiale E_c, perdita per ionizzazione dEx_0 e il passo di avanzamento $s \in [0, 1]$. La procedura è stata la seguente:

- si sono selezionati due materiali diversi: acqua liquida e silicato di bismuto.
- per uno stesso materiale si sono eseguite più simulazioni variando E_0 .

Struttura del progetto

Scelta dei parametri

I valori scelti dall'utente sono i seguenti: energia della particella iniziale E_0, energia critica del materiale E_c, perdita per ionizzazione dEx_0 e il passo di avanzamento $s \in [0, 1]$. La procedura è stata la seguente:

- si sono selezionati due materiali diversi: acqua liquida e silicato di bismuto.
- per uno stesso materiale si sono eseguite più simulazioni variando E_0 .
- si analizzano i risultati mediante via grafica e dai valori restuiti dal programma.

Struttura del progetto

Scelta dei parametri

I valori scelti dall'utente sono i seguenti: energia della particella iniziale E_0, energia critica del materiale E_c, perdita per ionizzazione dEx_0 e il passo di avanzamento $s \in [0, 1]$. La procedura è stata la seguente:

- si sono selezionati due materiali diversi: acqua liquida e silicato di bismuto.
- per uno stesso materiale si sono eseguite più simulazioni variando E_0 .
- si analizzano i risultati mediante via grafica e dai valori restuiti dal programma.

Grandezze nel corso dello sciame

Step	# elettroni	# fotoni	# positroni	# particelle totali	Energia persa (MeV)
0	1	0	0	1	8478.655494348224
1	1	0	0	1	8478.655494348224
2	1	0	0	1	8478.655494348224
3	1	0	0	1	8478.655494348224
4	1	0	0	1	8478.655494348224
5	1	0	0	1	8478.655494348224
6	1	1	0	2	8478.655494348224
7	1	1	0	2	8478.655494348224
8	1	2	0	3	8478.655494348224
9	1	3	0	4	8478.655494348224
10	2	2	1	5	8478.655494348224
11	3	2	2	7	25435.96648304467
12	4	3	3	10	42393.27747174112
13	2	5	1	8	31864.01853453528
14	3	5	2	10	25435.96648304467
15	2	4	1	7	21440.076537494726
16	4	4	3	11	25435.96648304467
17	1	5	1	7	22824.330606226024
18	1	6	1	8	16957.31098869645
19	5	2	5	12	16957.31098869645
20	3	2	3	8	54550.27532522272
21	2	2	2	6	34861.756142971455
22	4	2	4	10	33914.6219773929
23	3	3	3	9	38860.70195353993
24	1	3	1	5	27694.838976645813
25	0	3	1	4	9756.640610390292
26	0	3	0	3	3331.118591902028
27	1	2	1	4	0
28	0	2	0	2	4080.4954231651936
29	0	2	0	2	0
30	1	1	1	3	0

Sviluppo dello sciame

Lo sviluppo dello sciame avviene attraverso la produzione di particelle definite attraverso le classi *Fotone*, *Elettrone* e *Positrone*. La particella iniziale che genera lo sciame viene scelta dall'utente. Nel metodo `simula_sciame` è stata creata una lista di dizionari in cui viene specificato il tipo di particella e la sua energia.

Sviluppo dello sciame

Lo sviluppo dello sciame avviene attraverso la produzione di particelle definite attraverso le classi *Fotone*, *Elettrone* e *Positrone*. La particella iniziale che genera lo sciame viene scelta dall'utente. Nel metodo `simula_sciame` è stata creata una lista di dizionari in cui viene specificato il tipo di particella e la sua energia.

- a partire dalla particella iniziale vengono controllate le condizioni sulla sua energia fino a che l'energia delle particella è positiva (ciclo `while` esterno).

Sviluppo dello sciame

Lo sviluppo dello sciame avviene attraverso la produzione di particelle definite attraverso le classi *Fotone*, *Elettrone* e *Positrone*. La particella iniziale che genera lo sciame viene scelta dall'utente. Nel metodo `simula_sciame` è stata creata una lista di dizionari in cui viene specificato il tipo di particella e la sua energia.

- a partire dalla particella iniziale vengono controllate le condizioni sulla sua energia fino a che l'energia delle particella è positiva (ciclo `while` esterno).
- dipendentemente dall'esito delle condizioni (stabilite tramite gli `if`) vengono prodotte le rispettive particelle creandole tramite le classi sopra definite.

Sviluppo dello sciame

Lo sviluppo dello sciame avviene attraverso la produzione di particelle definite attraverso le classi *Fotone*, *Elettrone* e *Positrone*. La particella iniziale che genera lo sciame viene scelta dall'utente. Nel metodo `simula_sciame` è stata creata una lista di dizionari in cui viene specificato il tipo di particella e la sua energia.

- a partire dalla particella iniziale vengono controllate le condizioni sulla sua energia fino a che l'energia delle particella è positiva (ciclo `while` esterno).
- dipendentemente dall'esito delle condizioni (stabilite tramite gli `if`) vengono prodotte le rispettive particelle creandole tramite le classi sopra definite.
- il ciclo `while` viene iterato fino a che non ci sono più particelle che possono depositare energia.

Sviluppo dello sciame

Lo sviluppo dello sciame avviene attraverso la produzione di particelle definite attraverso le classi *Fotone*, *Elettrone* e *Positrone*. La particella iniziale che genera lo sciame viene scelta dall'utente. Nel metodo `simula_sciame` è stata creata una lista di dizionari in cui viene specificato il tipo di particella e la sua energia.

- a partire dalla particella iniziale vengono controllate le condizioni sulla sua energia fino a che l'energia delle particella è positiva (ciclo `while` esterno).
- dipendentemente dall'esito delle condizioni (stabilite tramite gli `if`) vengono prodotte le rispettive particelle creandole tramite le classi sopra definite.
- il ciclo `while` viene iterato fino a che non ci sono più particelle che possono depositare energia.
- la lista delle particelle viene aggiornata ad ogni step e tramite il ciclo `for` che agisce sulla lista si controllano le suddette condizioni.

Struttura del codice

- Classi per la definizione delle particelle.

Struttura del codice

- Classi per la definizione delle particelle.
- Metodo di inizializzazione della classe.

Struttura del codice

- Classi per la definizione delle particelle.
- Metodo di inizializzazione della classe.

```
class Fotone:
"""classe che definisce una particella
     di tipo fotone
    specificandone l'energia """

    def __init__(self, energia):
        self.energia = energia
class Elettrone:
  """classe che definisce una
     particella di tipo elettrone
    specificandone l'energia """

    def __init__(self, energia):
        self.energia = energia
class Positrone:
  """classe che definisce una
     particella di tipo positrone
    specificandone l'energia """

    def __init__(self, energia):
        self.energia = energia
```

Struttura del codice

- Classi per la definizione delle particelle.
- Metodo di inizializzazione della classe.

```
class Fotone:
"""classe che definisce una particella
     di tipo fotone
    specificandone l'energia """

    def __init__(self, energia):
        self.energia = energia
class Elettrone:
  """classe che definisce una
     particella di tipo elettrone
    specificandone l'energia """

    def __init__(self, energia):
        self.energia = energia
class Positrone:
  """classe che definisce una
     particella di tipo positrone
    specificandone l'energia """

    def __init__(self, energia):
        self.energia = energia
```

```
def __init__(self,tipo_particella, E0,
     Ec_elettroni, Ec_positroni, dEx0,
     s, X0):
        self._tipo_particella =
     tipo_particella
        self._E0 = E0
        self._Ec_elettroni =
     Ec_elettroni
        self._Ec_positroni =
     Ec_positroni
        self._dEx0 = dEx0
        self._s = s
        self._X0 = X0
        self.particelle = [{'tipo':
     self._tipo_particella, 'energia':
     E0}]
        self.energie_persa_per_step =
     []
        self.numero_particelle_per_step
      = []
        self.array_passi = []
```

Struttura del codice

```
while (energia_particella > 0):
            controllo = []
                    # lista per
    controllare l'arresto dello sciame
            step += 1
                    # lo step viene
    incrementato di 1
            distanza= step*self._s
                    # calcolo della
    distanza percorsa
            array_passi.append(step)
            array_distanze.append(
    distanza)
            nuove_particelle = []
                    # lista che viene
    aggiornata ad ogni iterazione

            n_elettroni = 0
            n_fotoni = 0
            n_positroni = 0
            energia_persa_per_step = 0
```

Struttura del codice

```
while (energia_particella > 0):
            controllo = []
                    # lista per
    controllare l'arresto dello sciame
            step += 1
                    # lo step viene
    incrementato di 1
            distanza= step*self._s
                    # calcolo della
    distanza percorsa
            array_passi.append(step)
            array_distanze.append(
    distanza)
            nuove_particelle = []
                    # lista che viene
    aggiornata ad ogni iterazione

            n_elettroni = 0
            n_fotoni = 0
            n_positroni = 0
            energia_persa_per_step = 0
```

- il ciclo viene iterato fino a che le particelle possono depositare energia.

Struttura del codice

```
while (energia_particella > 0):
            controllo = []
                    # lista per
    controllare l'arresto dello sciame
            step += 1
                    # lo step viene
    incrementato di 1
            distanza= step*self._s
                    # calcolo della
    distanza percorsa
            array_passi.append(step)
            array_distanze.append(
    distanza)
            nuove_particelle = []
                    # lista che viene
    aggiornata ad ogni iterazione

            n_elettroni = 0
            n_fotoni = 0
            n_positroni = 0
            energia_persa_per_step = 0
```

- il ciclo viene iterato fino a che le particelle possono depositare energia.
- la lista `controllo` permette l'arresto dello sciame.

Struttura del codice

```
while (energia_particella > 0):
            controllo = []
                    # lista per
    controllare l'arresto dello sciame
            step += 1
                    # lo step viene
    incrementato di 1
            distanza= step*self._s
                    # calcolo della
    distanza percorsa
            array_passi.append(step)
            array_distanze.append(
    distanza)
            nuove_particelle = []
                    # lista che viene
    aggiornata ad ogni iterazione

            n_elettroni = 0
            n_fotoni = 0
            n_positroni = 0
            energia_persa_per_step = 0
```

- il ciclo viene iterato fino a che le particelle possono depositare energia.
- la lista `controllo` permette l'arresto dello sciame.
- ad ogni step la lista `nuove_particelle` viene aggiornata.

Struttura del codice

```
while (energia_particella > 0):
            controllo = []
                    # lista per
    controllare l'arresto dello sciame
            step += 1
                    # lo step viene
    incrementato di 1
            distanza= step*self._s
                    # calcolo della
    distanza percorsa
            array_passi.append(step)
            array_distanze.append(
    distanza)
            nuove_particelle = []
                    # lista che viene
    aggiornata ad ogni iterazione

            n_elettroni = 0
            n_fotoni = 0
            n_positroni = 0
            energia_persa_per_step = 0
```

- il ciclo viene iterato fino a che le particelle possono depositare energia.
- la lista `controllo` permette l'arresto dello sciame.
- ad ogni step la lista `nuove_particelle` viene aggiornata.
- si tiene conto del numero delle particelle e dell'energia ad ogni step.

Struttura codice

- Di seguito si riporta il controllo delle condizioni sull'energia per una particella di tipo elettrone

Struttura codice

- Di seguito si riporta il controllo delle condizioni sull'energia per una particella di tipo elettrone

```
if tipo_particella == 'Elettrone':
                    if energia_particella >
    (self._dEx0*self._s):
                        perdita_energia =
    self._dEx0*self._s

    energia_persa_per_step +=
    perdita_energia
                        nuova_energia =
    energia_particella - perdita_energia
                        n_elettroni += 1
                        if nuova_energia >
    self._Ec_elettroni:
                            probabilita = 1-
    np.exp(-self._s)
                            if (np.random.
    uniform() < probabilita):

    nuove_particelle.append({'tipo': '
    Fotone', 'energia': nuova_energia/2})

    nuove_particelle.append({'tipo': '
    Elettrone', 'energia': nuova_energia
    /2})
```

Struttura codice

- Di seguito si riporta il controllo delle condizioni sull'energia per una particella di tipo elettrone

```
if tipo_particella == 'Elettrone':
                    if energia_particella >
    (self._dEx0*self._s):
                        perdita_energia =
    self._dEx0*self._s

    energia_persa_per_step +=
    perdita_energia
                        nuova_energia =
    energia_particella - perdita_energia
                        n_elettroni += 1
                        if nuova_energia >
    self._Ec_elettroni:
                            probabilita = 1-
    np.exp(-self._s)
                            if (np.random.
    uniform() < probabilita):

    nuove_particelle.append({'tipo': '
    Fotone', 'energia': nuova_energia/2})

    nuove_particelle.append({'tipo': '
    Elettrone', 'energia': nuova_energia
    /2})
```

```
    n_fotoni +=1
    b.append(nuova_energia/2)
    b.append(nuova_energia/2)
     else:
        nuove_particelle.append({'tipo': '
     Elettrone', 'energia': nuova_energia})
                                    b.append(
     nuova_energia)
    else:
                            energia_depositata =
      np.random.uniform(low = 0, high=
     energia_particella)

     energia_persa_per_step +=
     energia_depositata
                            continue
```

Struttura codice

- L'arresto dello sciame avviene come segue:

Struttura codice

- L'arresto dello sciame avviene come segue:

```
for i in b:
                    if i > 0:
                        controllo.append(1)
```

Struttura codice

- L'arresto dello sciame avviene come segue:

```
for i in b:
                    if i > 0:
                        controllo.append(1)
```

```
if len(controllo)==0:
                check = 1
if check ==1:
        break
```

Struttura codice

- L'arresto dello sciame avviene come segue:

```
for i in b:
                    if i > 0:
                        controllo.append(1)
```

```
if len(controllo)==0:
                check = 1
if check ==1:
        break
```

```
energia_corrente -= energia_persa_per_step
self.particelle = nuove_particelle
self.numero_particelle_per_step.append(len(
    self.particelle))
energie.append(energia_persa_per_step)
n_elettroni_totali.append(n_elettroni)
n_fotoni_totali.append(n_fotoni)
n_positroni_totali.append(n_positroni)
somma_particelle.append(n_elettroni+n_fotoni
    +n_positroni)
array_energia_corrente.append(
    energia_corrente)
```

Struttura codice

- L'arresto dello sciame avviene come segue:

```
for i in b:
                    if i > 0:
                        controllo.append(1)
```

```
if len(controllo)==0:
                check = 1
if check ==1:
        break
```

```
energia_corrente -= energia_persa_per_step
self.particelle = nuove_particelle
self.numero_particelle_per_step.append(len(
    self.particelle))
energie.append(energia_persa_per_step)
n_elettroni_totali.append(n_elettroni)
n_fotoni_totali.append(n_fotoni)
n_positroni_totali.append(n_positroni)
somma_particelle.append(n_elettroni+n_fotoni
    +n_positroni)
array_energia_corrente.append(
    energia_corrente)
```

- b è la lista contenente le energie delle particelle.

Struttura codice

- L'arresto dello sciame avviene come segue:

```
for i in b:
                    if i > 0:
                        controllo.append(1)
```

```
if len(controllo)==0:
                check = 1
if check ==1:
        break
```

```
energia_corrente -= energia_persa_per_step
self.particelle = nuove_particelle
self.numero_particelle_per_step.append(len(
    self.particelle))
energie.append(energia_persa_per_step)
n_elettroni_totali.append(n_elettroni)
n_fotoni_totali.append(n_fotoni)
n_positroni_totali.append(n_positroni)
somma_particelle.append(n_elettroni+n_fotoni
    +n_positroni)
array_energia_corrente.append(
    energia_corrente)
```

- b è la lista contenente le energie delle particelle.
- tramite la lista `controllo` si verifica l'arresto dello sciame.

Struttura codice

- L'arresto dello sciame avviene come segue:

```
for i in b:
                    if i > 0:
                        controllo.append(1)
```

```
if len(controllo)==0:
                check = 1
if check ==1:
        break
```

```
energia_corrente -= energia_persa_per_step
self.particelle = nuove_particelle
self.numero_particelle_per_step.append(len(
     self.particelle))
energie.append(energia_persa_per_step)
n_elettroni_totali.append(n_elettroni)
n_fotoni_totali.append(n_fotoni)
n_positroni_totali.append(n_positroni)
somma_particelle.append(n_elettroni+n_fotoni
     +n_positroni)
array_energia_corrente.append(
     energia_corrente)
```

- b è la lista contenente le energie delle particelle.
- tramite la lista `controllo` si verifica l'arresto dello sciame.
- se la condizione dell' `if` è verificata, il ciclo si arresta dato che non ci sono più particelle che depositano energia.

Struttura codice

- L'arresto dello sciame avviene come segue:

```
for i in b:
                    if i > 0:
                        controllo.append(1)
```

```
if len(controllo)==0:
                check = 1
if check ==1:
        break
```

```
energia_corrente -= energia_persa_per_step
self.particelle = nuove_particelle
self.numero_particelle_per_step.append(len(
    self.particelle))
energie.append(energia_persa_per_step)
n_elettroni_totali.append(n_elettroni)
n_fotoni_totali.append(n_fotoni)
n_positroni_totali.append(n_positroni)
somma_particelle.append(n_elettroni+n_fotoni
    +n_positroni)
array_energia_corrente.append(
    energia_corrente)
```

- b è la lista contenente le energie delle particelle.
- tramite la lista `controllo` si verifica l'arresto dello sciame.
- se la condizione dell' `if` è verificata, il ciclo si arresta dato che non ci sono più particelle che depositano energia.
- ad ogni step le liste introdotte vengono aggiornate.

Indice

Simulazione dello sciame

Si riportano ora i risultati ottenuti dalla simulazione partendo da un elettrone con energia iniziale $E_0 = 600\text{GeV}$ e $s = 0.33$ per i due materiali e per i valori scelti dall'utente. Per brevità si riportano i grafici relativi allo sviluppo longitudinale: numero di particelle e energia depositata.

Simulazione dello sciame

Si riportano ora i risultati ottenuti dalla simulazione partendo da un elettrone con energia iniziale $E_0 = 600\text{GeV}$ e $s = 0.33$ per i due materiali e per i valori scelti dall'utente. Per brevità si riportano i grafici relativi allo sviluppo longitudinale: numero di particelle e energia depositata.

- Per i valori scelti dall'utente si hanno i seguenti risultati:

Simulazione dello sciame

Si riportano ora i risultati ottenuti dalla simulazione partendo da un elettrone con energia iniziale $E_0 = 600\text{GeV}$ e $s = 0.33$ per i due materiali e per i valori scelti dall'utente. Per brevità si riportano i grafici relativi allo sviluppo longitudinale: numero di particelle e energia depositata.

- Per i valori scelti dall'utente si hanno i seguenti risultati:

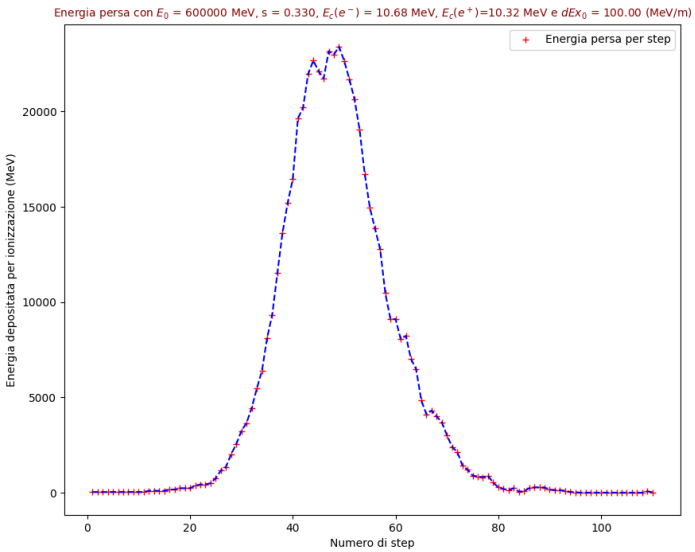


Figure: Energia persa per step

Simulazione dello sciame

Si riportano ora i risultati ottenuti dalla simulazione partendo da un elettrone con energia iniziale $E_0 = 600\text{GeV}$ e $s = 0.33$ per i due materiali e per i valori scelti dall'utente. Per brevità si riportano i grafici relativi allo sviluppo longitudinale: numero di particelle e energia depositata.

- Per i valori scelti dall'utente si hanno i seguenti risultati:

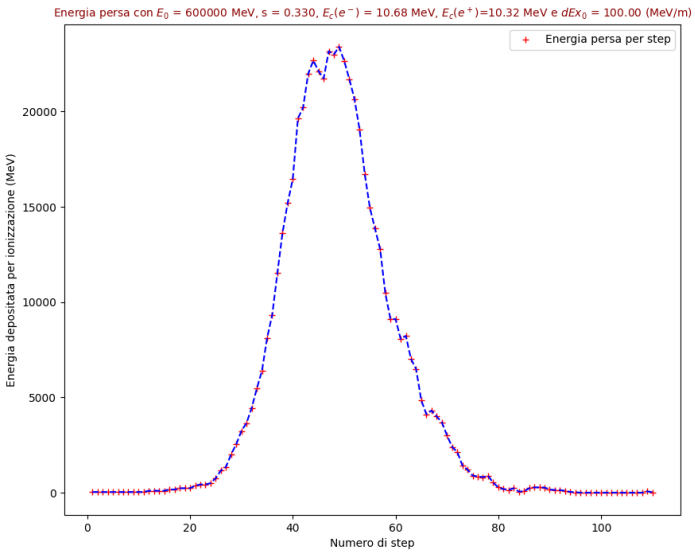


Figure: Energia persa per step

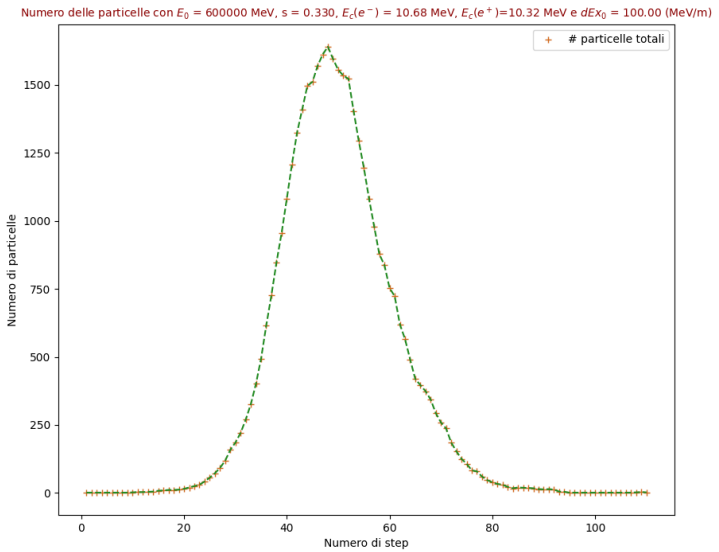


Figure: Numero particelle per step

Simulazione dello sciame

- I risultati per i due materiali sono i seguenti:

Simulazione dello sciame

- I risultati per i due materiali sono i seguenti:

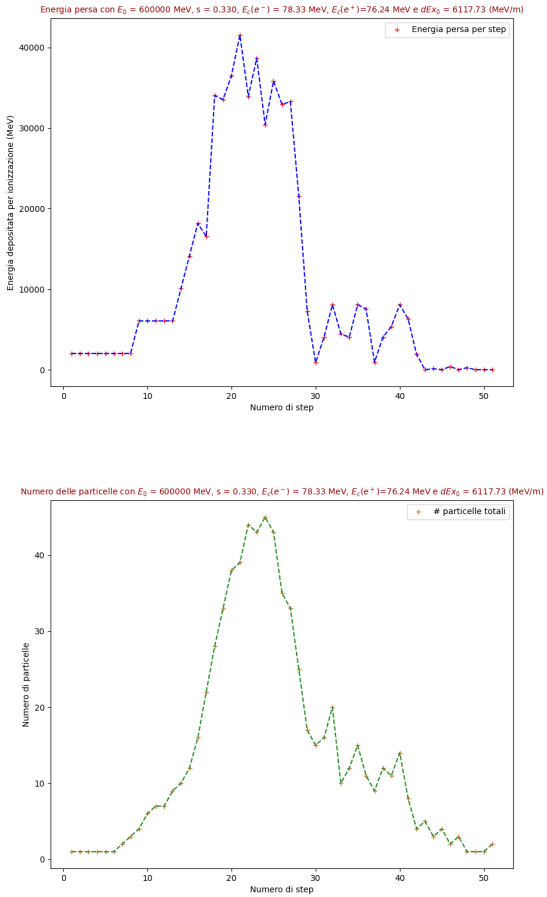
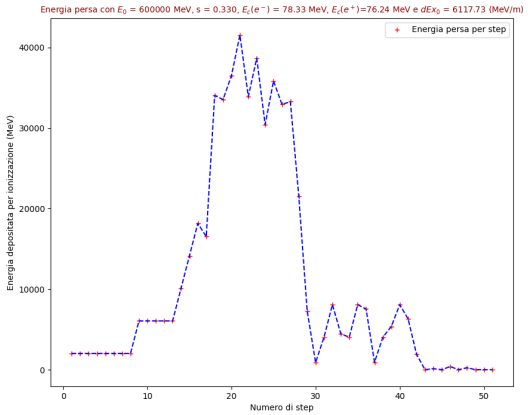


Figure: Risultati per H_2O

Simulazione dello sciame

- I risultati per i due materiali sono i seguenti:



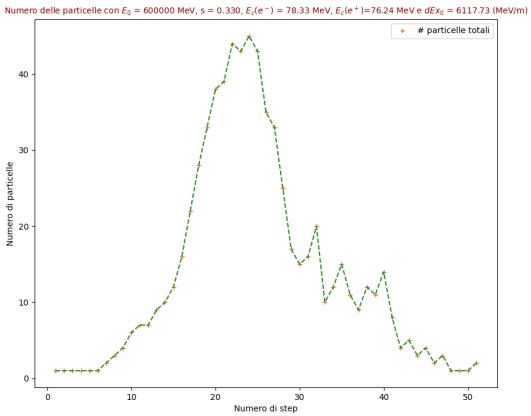
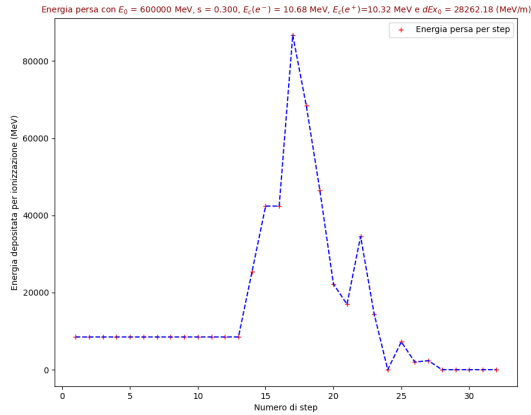


Figure: Risultati per H_2O



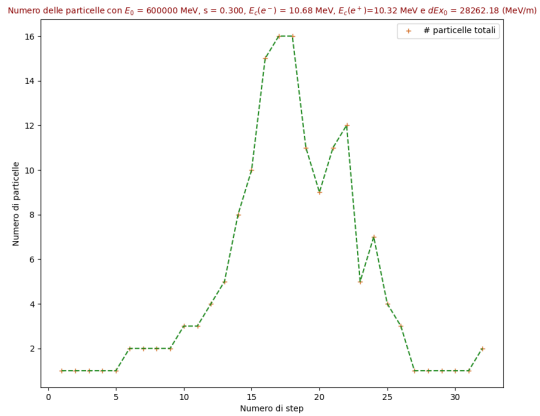


Figure: Risultati per BSO

Confronto tra simulazioni con energia diversa

- Riportiamo ora il confronto tra simulazioni a partire da energie diverse per i due materiali: $E_0 = 600\text{GeV}, 700\text{GeV}, 800\text{GeV}$ e $s = 0.55$.

Confronto tra simulazioni con energia diversa

- Riportiamo ora il confronto tra simulazioni a partire da energie diverse per i due materiali: $E_0 = 600\text{GeV}, 700\text{GeV}, 800\text{GeV}$ e $s = 0.55$.

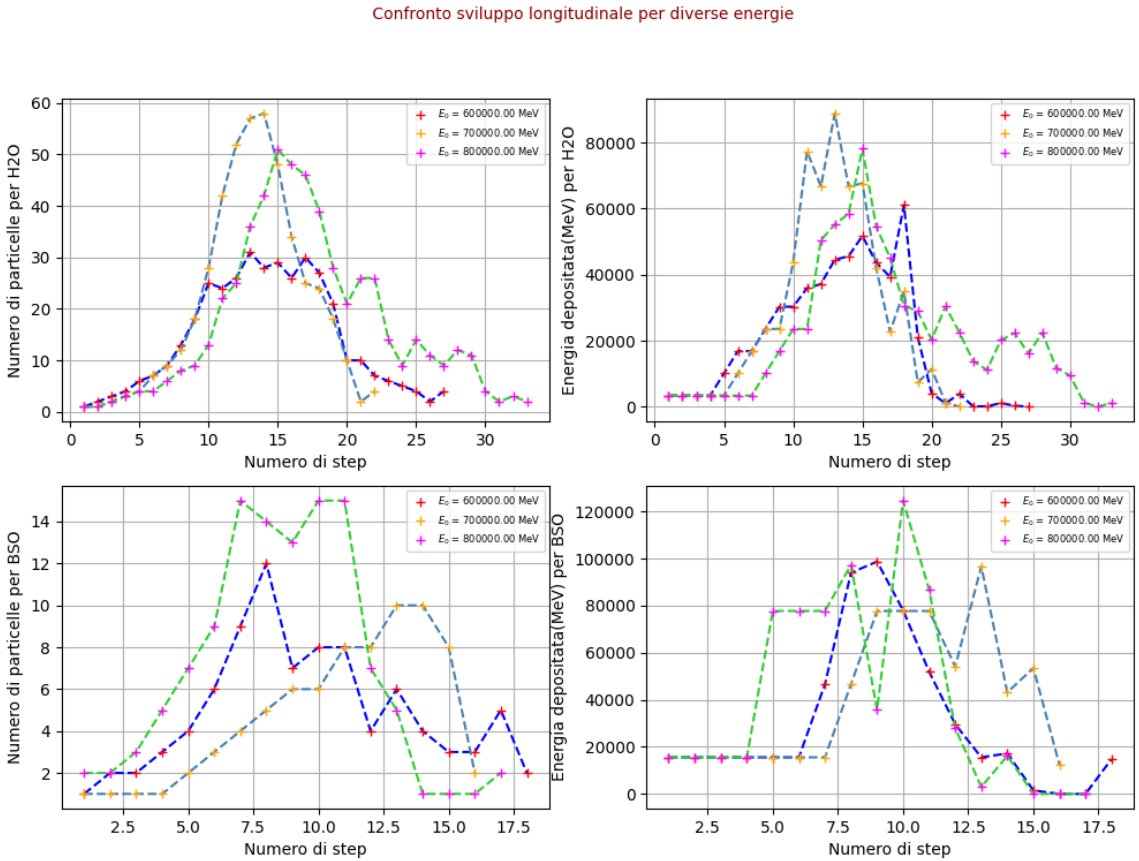


Figure: Confronto simulazioni

Confronto tra simulazioni con energia diversa

- Ripetiamo ora il confronto tra simulazioni a partire dalle stesse energia cambiando il passo di avanzamento, $s = 0.7$.

Confronto tra simulazioni con energia diversa

- Ripetiamo ora il confronto tra simulazioni a partire dalle stesse energia cambiando il passo di avanzamento, $s = 0.7$.

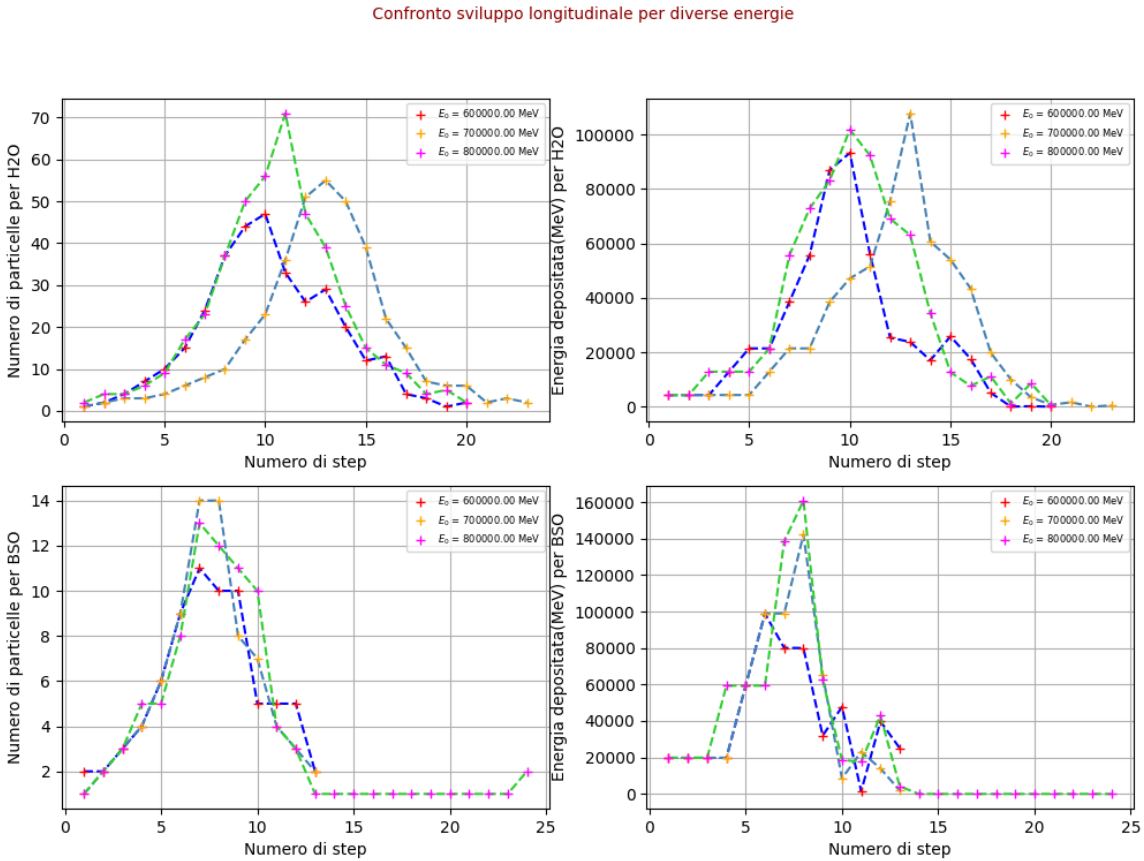


Figure: Confronto simulazioni

Andamento dell'energia

- Riportiamo poi l'andamento dell'energia per i due materiali con i parametri: $E_0 = 600\text{GeV}$ e $s = 0.33$.

Andamento dell'energia

- Riportiamo poi l'andamento dell'energia per i due materiali con i parametri: $E_0 = 600\text{GeV}$ e $s = 0.33$.

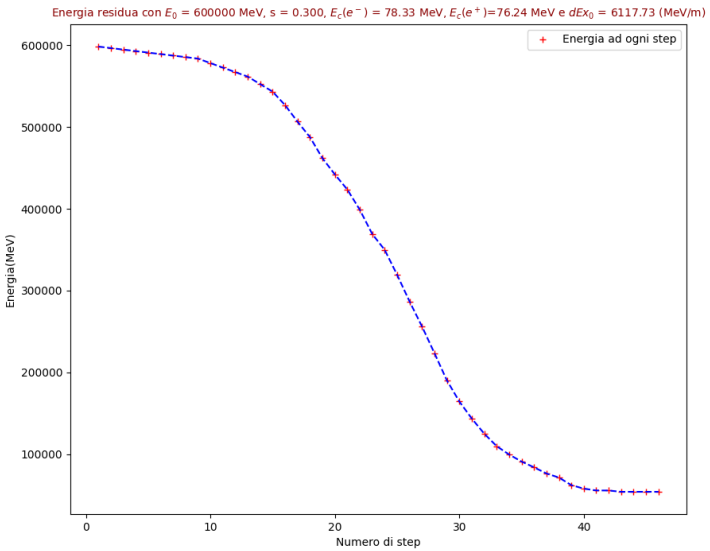


Figure: Andamento energia per H_2O

Andamento dell'energia

- Riportiamo poi l'andamento dell'energia per i due materiali con i parametri: $E_0 = 600\text{GeV}$ e $s = 0.33$.

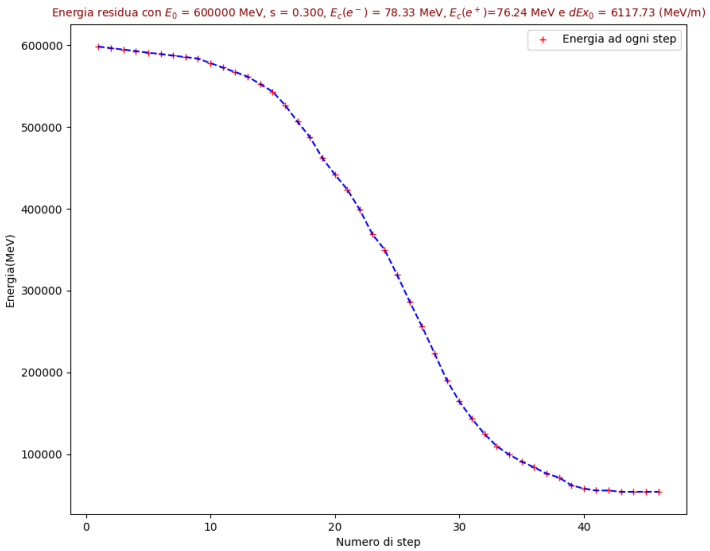


Figure: Andamento energia per H_2O

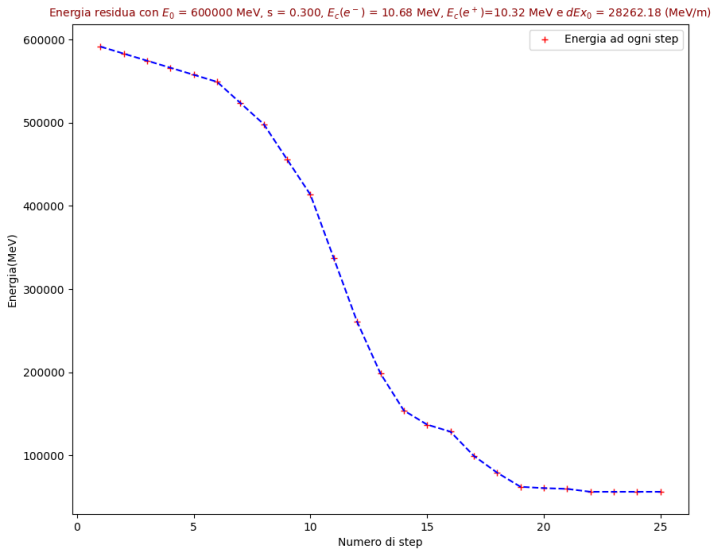


Figure: Andamento energia per BSO