Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea Magistrale in Matematica

RETI NEURALI "PHYSICS-INFORMED" PER MODELLI EPIDEMIOLOGICI

Tesi di Laurea in Analisi Numerica

Relatore:
Chiar.ma Prof.ssa
Fabiana Zama

Presentata da:
Irene Migliore

Correlatori:
Chiar.ma Prof.ssa
Elena Loli Piccolomini

Dott. Andrea Sebastiani

Sessione Unica
Anno Accademico 2020/2021

Ai miei genitori Lillo e Maria
e a mia sorella Ilaria

Indice

Introduzione

La malattia COVID-19 associata alla sindrome respiratoria acuta grave da coronavirus 2 (SARS-CoV-2) ha rappresentato una grave minaccia per la salute pubblica globale sin dalla sua scoperta in Cina, nel dicembre del 2019. L'11 marzo 2020 l'Organizzazione Mondiale di Sanità (OMS) ha caratterizzato la malattia COVID-19 come pandemia [1] e il mese successivo l'Organizzazione delle Nazioni Unite (ONU) ha definito questa pandemia una crisi, oltre che sanitaria, umana, ecologica e sociale [2]. I governi di diversi Paesi hanno introdotto misure restrittive per limitare la trasmissione del virus. In seguito, la scoperta di vaccini efficaci per il COVID-19 e l'inizio delle somministrazioni a fine 2020, hanno rallentato la diffusione della malattia, permesso un allentamento delle restrizioni, diminuendo il numero di soggetti ospedalizzati a causa del virus. A marzo 2022 l'epidemia si trova in una fase avanzata e l'umanità ha imparato a convivere con il virus. Tale fase è caratterizzata da numerose varianti del virus e da una visione più ampia sullo sviluppo della malattia grazie ai numerosi archivi di dati messi a disposizione dai governi e da vari enti. Gli studiosi hanno effettuato numerosi studi e pubblicato più di 100000 articoli ed in particolar modo l'applicazione di modelli epidemiologici costruiti a partire dai dati raccolti, ha permesso la previsione di diversi scenari sullo sviluppo della malattia, nel breve-medio termine.

Gli obiettivi di questa tesi sono:

1. la ricerca dei principali archivi di riferimento nazionali e mondiali sulla malattia COVID-19, quindi l'estrazione, la lettura e l'analisi di tali dati, con particolare attenzione ai dati sulle varianti del virus e sui vaccini;

2. l'esplorazione di modelli deterministici, in particolare quelli epidemiologici compartimentali esistenti che tengono conto delle vaccinazioni ed eventualmente ricorrono alla conoscenza delle varianti per spiegare il comportamento del modello;

3. l'utilizzo di un nuovo approccio basato sul *deep learning* che include i modelli epidemiologici in reti neurali, per questo note come "physics - informed" (PINNs), allenate su dati reali.

Un primo contributo di questo lavoro è la realizzazione di molteplici script Python che consentono l'acquisizione automatica di dati epidemiologici da diversi database open source presenti su Github, l'analisi e la visualizzazione di tali dati con l'ausilio delle librerie *pandas* e *matplotlib.pyplot*. Successivamente, si è passato allo studio nel dettaglio delle PINNs, recentemente introdotte nell'articolo [3], e alla loro applicazione al complesso modello compartimentale SEIJDHR [4]. Un ultimo contributo proviene dal test di alcuni codici Python tratti da [5] con dati in input riguardanti la città di New York e dall'adattamento di tali codici ai dati sulla realtà italiana.

I risultati ottenuti dalle sperimentazioni mostrano delle previsioni abbastanza fedeli ai dati reali, tranne per una sottostima dei nuovi positivi nel periodo di sviluppo della variante Omicron.

La tesi è strutturata in quattro capitoli come segue:

- nel primo vengono presentati i classici modelli compartimentali per le epidemie, un recente modello (SEIJDHR) che tiene conto degli effetti delle vaccinazioni e una sua variazione in cui viene considerato il ritardo di risposta;

- il secondo tratta delle reti neurali artificiali: dal percettrone alle reti neurali profonde fino alla reti neurali "physics-informed" e alla loro applicazione al modello SEIJDHR;

- nel terzo vengono introdotti gli strumenti utilizzati per leggere ed estrarre i dati e viene poi data una panoramica sui dati disponibili sulla malattia COVID-19, accessibili a tutti e non, con particolare riguardo ai dati italiani e newyorkesi;

- infine, nel quarto vengono riportati frammenti rilevanti dei codici utilizzati e i risultati numerici ottenuti applicando le PINNs al modello SEIJDHR. In particolare vengono mostrati i grafici sulle previsioni nel breve-medio termine, ottenuti dando in input dati sul numero di positivi, ospedalizzati e deceduti giornalieri prima rigurdanti la città di New York e poi l'Italia.

Capitolo 1

Modelli compartimentali per le epidemie

In questo capitolo presenteremo molteplici modelli compartimentali epidemiologici. Daremo dapprima una panoramica dei modelli classici per poi soffermarci su un modello che tiene conto degli effetti delle vaccinazioni ed una sua variazione in cui viene considerato il *ritardo di risposta*.
Le malattie infettive possono essere descritte matematicamente utilizzando modelli statistici, deterministici o stocastici [6]. I modelli compartimentali fanno parte dei modelli deterministici e si basano sulla suddivisione di una popolazione di individui in compartimenti. Per **compartimento** si intende lo stato epidemiologico nel quale un individuo si trova durante il manifestarsi di una malattia infettiva. Ad esempio un individuo che ha contratto la malattia da virus Ebola (EVD) è definito *infetto* e, per tutto il periodo dell'infezione, farà parte del *compartimento degli infetti*.

Per semplicità, assumiamo che la popolazione sia fissa, composta da $N > 0$ individui e consideriamo solo modelli compartimentali costruiti con sistemi non lineari di equazioni differenziali ordinarie (ODEs), nella variabile tempo.

1.1 Modelli classici

Presentiamo i quattro modelli più utilizzati per descrivere la diffusione di una malattia infettiva.

1.1.1 SIR

Nel 1927 William Kermack e Anderson McKendrick hanno proposto il cosiddetto modello SIR [7], acronimo di *Susceptible*, *Infected* e *Recovered* (o

Removed). La popolazione viene infatti suddivisa in tre compartimenti S, I, R, mostrati in figura 1.1.

Figura 1.1: Diagramma di trasferimento per il modello (1.2)

Fissato il tempo $t \geq 0$, viene indicata con:

- $S(t)$ la parte di popolazione suscettibile, ossia non infetta ma che può essere infettata;
- $I(t)$ la parte di popolazione infetta e contagiosa;
- $R(t)$ la parte di popolazione che non può diffondere la malattia; comprende gli individui in quarantena, i guariti e i deceduti.

La transizione di un individuo da un compartimento all'altro può avvenire nella sola direzione indicata dalle frecce in figura 1.1 ed è regolata da due parametri non negativi:

1. β, chiamato tasso di trasmissione; esso incorpora il tasso di incontro tra individui suscettibili e infettivi insieme alla probabilità di trasmissione.
2. γ, chiamato tasso di rimozione o di recupero; il suo reciproco $\frac{1}{\gamma}$ determina il periodo infettivo medio.

Trascurando i nuovi nati, i deceduti per altre cause e le migrazioni si ha:

$$S(t) + I(t) + R(t) = N \tag{1.1}$$

Il modello è descritto dal sistema di ODE [6]:

$$\begin{cases} \frac{dS(t)}{dt} = -\frac{\beta}{N} S(t) I(t), \\ \frac{dI(t)}{dt} = \frac{\beta}{N} S(t) I(t) - \gamma I(t), \\ \frac{dR(t)}{dt} = \gamma I(t). \end{cases} \tag{1.2}$$

con condizioni iniziali:

$$S(0) = S_0 > 0,\ I(0) = I_0 > 0,\ R(0) = 0 \tag{1.3}$$

Sostituendo $t = 0$ in (1.1) e le condizioni (1.3) $\Rightarrow S_0 = N - I_0$.

Osservazione 1. Dalla prima equazione del sistema (1.2) otteniamo

$$\frac{dS(t)}{dt} \leq 0, \forall t \geq 0 \tag{1.4}$$

da cui segue che $S(t)$ è sempre decrescente. In particolare:

$$S(t) \leq S_0, \forall t \geq 0 \tag{1.5}$$

Dalla seconda, invece, troviamo una condizione sull'inizio della diffusione dell'epidemia, che avviene se:

$$\frac{dI(t)}{dt} > 0,\ t \in [0, \hat{t}),\ \text{per qualche}\, \hat{t} \tag{1.6}$$

ovvero se:

$$I(t)\left(\frac{\beta}{N}S(t) - \gamma\right) > 0,\ t \in [0, \hat{t}) \tag{1.7}$$

e dunque

$$S(t) > \frac{\gamma}{\beta}N,\ t \in [0, \hat{t}) \tag{1.8}$$

Combinando la (1.5) e (1.8) si ha:

$$S_0 > \frac{\gamma}{\beta}N \tag{1.9}$$

Supponendo infine che all'inizio di un'infezione $S_0 = N - 1$, con $N >> 1$ si ha che:

$$\frac{1}{N} \sim 0 \Rightarrow \frac{N-1}{N} \sim 1 > \frac{\gamma}{\beta} \tag{1.10}$$

Diekmann, Heesterbeek e Metz [8], nel 1990, diedero la seguente definizione di uno degli indicatori di base in epidemiologia, accreditata dalla comunità scientifica:

Definizione 1.1. Il **numero di riproduzione di base**, indicato con R_0, è il numero medio di infezioni secondarie prodotte da un tipico individuo infetto durante tutto il suo periodo di infettività in una popolazione completamente suscettibile, cioè mai venuta a contatto con il nuovo patogeno emergente.

Tale indicatore è divenuto noto soprattutto negli anni cinquanta, quando è stato usato per studiare la propagazione della malaria [9]. Per il modello SIR si trova considerando il reciproco in (1.10):

$$R_0 = \frac{\beta}{\gamma} \tag{1.11}$$

ed è il prodotto tra il tasso di trasmissione β e il periodo infettivo medio, $1/\gamma$.

Osservazione 2. R_0 viene detto *soglia critica* in quanto si verifica il **fenomeno di soglia**:

1. Se $R_0 > 1 \Rightarrow$ l'epidemia inizia a diffondersi;
2. Se $R_0 < 1 \Rightarrow$ il processo si estingue.

Osservazione 3. R_0 è utile solo all'inizio dell'infezione. Una popolazione raramente sarà totalmente suscettibile a un'infezione nel mondo reale. Alcuni contatti saranno immuni per un certo periodo di tempo, per esempio a causa di un'infezione precedente. Si utilizza quindi un altro indicatore:

Definizione 1.2. Indicato con R, R^* o R_t, il **numero di riproduzione effettivo** è il numero medio di casi secondari per caso infettivo in una popolazione composta da ospiti suscettibili e non suscettibili.

1.1.2 SEIR

Un individuo esposto al virus, prima di diventare infetto, attraversa la cosiddetta "fase di incubazione". Indicato con E il compartimento che comprende gli individui 'esposti' al virus ma non ancora infetti:

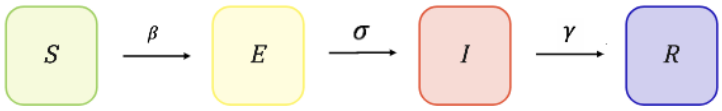

Figura 1.2: Diagramma di trasferimento per il modello (1.12)

e detto σ il tasso di incubazione, si ottiene il modello SEIR:

$$\begin{cases} \frac{dS(t)}{dt} = -\frac{\beta}{N} I(t)S(t), \\ \frac{dE(t)}{dt} = \frac{\beta}{N} I(t)S(t) - \sigma E(t), \\ \frac{dI(t)}{dt} = \sigma E(t) - \gamma I(t), \\ \frac{dR(t)}{dt} = \gamma I(t). \end{cases} \tag{1.12}$$

Osservazione 4. Dalle ODEs in (1.12) possiamo dedurre che $R_0 = \frac{\beta}{\gamma}$. Infatti, in questo caso, l'epidemia inizia a diffondersi se

$$\frac{dE(t)}{dt} + \frac{dI(t)}{dt} > 0 \iff \frac{\beta}{N} I(t)S(t) - \gamma I(t) > 0 \tag{1.13}$$

Ripetendo quanto visto in 1 si trova il numero di riproduzione di base del modello SEIR. Notiamo che è lo stesso valore trovato nel modello precedente.

1.1.3 SIRD

Nel precedente modello SIR (1.2), si assume che gli indivudui infetti, dopo un certo periodo di tempo, transitano nel compartimento R. Aggiungendo il compartimento D, rappresentante la parte di popolazione deceduta a causa dell'infezione e supponendo che R sia composto dagli individui in quarantena e guariti, si ottengono i quattro compartimenti mostrati di seguito:

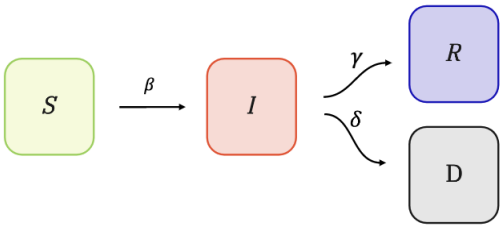

Figura 1.3: Diagramma di trasferimento per il modello (1.14)

Il modello SIRD basato su tale suddivisione della popolazione è descritto dal sistema di ODE [10]:

$$\begin{cases} \frac{dS(t)}{dt} = -\frac{\beta}{N} I(t)S(t), \\ \frac{dI(t)}{dt} = \frac{\beta}{N} I(t)S(t) - \gamma I(t) - \delta I(t), \\ \frac{dR(t)}{dt} = \gamma I(t), \\ \frac{dD(t)}{dt} = \delta I(t). \end{cases} \tag{1.14}$$

dove δ è il tasso di mortalità. Riscrivendo la seconda equazione del sistema (1.14) in

$$\frac{dI(t)}{dt} = \frac{\beta}{N} I(t)S(t) - (\gamma + \delta)I(t) \tag{1.15}$$

e seguendo i passaggi effettuati nell'osservazione 1 troviamo il numero di riproduzione di base per il modello SIRD:

$$R_0 = \frac{\beta}{\gamma + \delta} \tag{1.16}$$

1.1.4 SEIRD

Aggiungendo al modello (1.14) il compartimento E, che rappresenta la parte di popolazione infetta ma non ancora contagiosa (fase di incubazione)

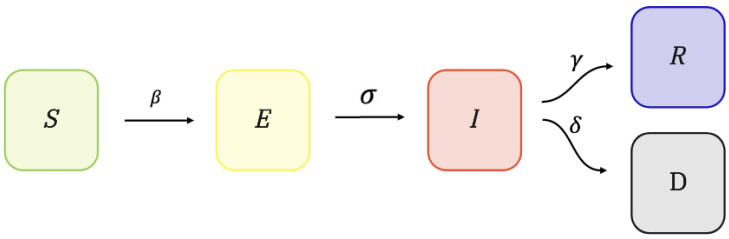

Figura 1.4: Diagramma di trasferimento per il modello (1.17)

otteniamo il modello SEIRD, descritto dal seguente sistema di ODEs:

$$\begin{cases} \frac{dS(t)}{dt} = -\frac{\beta}{N} I(t)S(t), \\ \frac{dE(t)}{dt} = \frac{\beta}{N} I(t)S(t) - \sigma E(t), \\ \frac{dI(t)}{dt} = \sigma E(t) - \gamma I(t) - \delta I(t), \\ \frac{dR(t)}{dt} = \gamma I(t), \\ \frac{dD(t)}{dt} = \delta I(t). \end{cases} \tag{1.17}$$

Analogamente a quanto visto nell'osservazione 4, per il modello (1.17),

$$R_0 = \frac{\beta}{\gamma + \delta} \tag{1.18}$$

1.2 Modello SEIJDHR

Un recente modello proposto nell'articolo [4] per descrivere la diffusione della malattia infettiva COVID-19, include due nuovi compartimenti, J e H e due compartimenti ausiliari I^c e H^c . Fissato il tempo t, viene indicata con:

- $J(t)$ la parte di popolazione asintomatica, ovvero che ha contratto il virus ma non mostra alcun sintomo;
- $H(t)$ la parte di popolazione ospedalizzata;
- $I^c(t)$ la totalità degli infetti dall'inizio della pandemia, chiamati infetti cumulativi;
- $H^c(t)$ la totalità degli ospedalizzati dall'inizio della pandemia, chiamati ospedalizzati cumulativi.

I compartimenti principali sono dunque S, E, I, J, D, H, R, schematizzati in figura 1.5 e da essi il modello prende il nome di SEIJDHR.

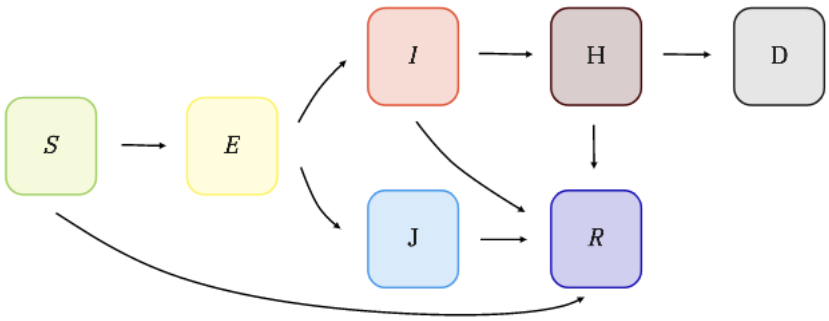

Figura 1.5: Diagramma di trasferimento per il modello (1.19)

Questo modello si differenzia dai precedenti, più che per l'introduzione di nuovi compartimenti, per la presenza di tre parametri dipendenti dal tempo:

1. $\beta_I(t)$, rappresenta il tasso di trasmissione comunitario, ovvero il tasso di infezione pro capite quando avviene un contatto;

2. $q(t)$, indica la percentuale di decessi correlati alla malattia dalla classe H;

3. $p(t)$, simboleggia la percentuale di persone ricoverate.

Tali parametri variano a seconda delle misure di controllo e delle politiche di salute pubblica attuate o, ancora, della scoperta di nuove terapie. In 2.4.1 mostreremo come trovarli attraverso reti neurali "physics-informed".

Si considera, inoltre, l'effetto delle vaccinazioni aggiungendo la connessione extra da S a R con tasso $\frac{V(t)S(t)}{N}$. $V(t)$ denota il numero di individui effettivamente vaccinati al tempo t. Il modello è descritto da un sistema di

nove ODEs:

$$\begin{cases} \frac{dS(t)}{dt} = -\frac{\beta_I(t)[I(t)+\epsilon J(t)]}{N} S(t) - \frac{V(t)S(t)}{N}, \\ \frac{dE(t)}{dt} = \frac{\beta_I(t)[I(t)+\epsilon J(t)]}{N} S(t) - \alpha E(t), \\ \frac{dI(t)}{dt} = \delta\alpha E(t) - \gamma I(t), \\ \frac{dJ(t)}{dt} = (1-\delta)\alpha E(t) - \gamma_a J(t), \\ \frac{dD(t)}{dt} = q(t)\phi_D H(t), \\ \frac{dH(t)}{dt} = p(t)\gamma I(t) - q(t)\phi_D H(t) - (1-q(t))\phi_R H(t), \\ \frac{dR(t)}{dt} = \gamma_a J(t) + (1-p(t))\gamma I(t) + (1-q(t))\phi_R H(t) + \frac{V(t)S(t)}{N}, \\ \frac{dI^c(t)}{dt} = \delta\alpha E(t), \\ \frac{dH^c(t)}{dt} = p(t)\gamma I(t). \end{cases} \tag{1.19}$$

I parametri del modello (1.19) vengono descritti in tabella 1.1.

PARAMETRO	**DESCRIZIONE**
ϵ	Rapporto d'infettività tra gli infetti lievi e gli infetti gravi
δ	Percentuale di infezioni asintomatiche
$1/\alpha$	Periodo medio di incubazione
$1/\gamma_a$	Periodo infettivo medio per la classe J
$1/\gamma$	Periodo infettivo sintomatico medio
$1/\phi_R$	Durata media in H prima del recupero della malattia
$1/\phi_D$	Durata media in H prima della morte per malattia

Tabella 1.1: Descrizione dei parametri del modello (1.19)

In appendice A riportiamo un modello proposto nell'articolo [11] in cui si tiene conto delle vaccinazioni direttamente nella definizione dei compartimenti.

1.2.1 Numero di riproduzione di controllo

Nel caso dell'introduzione di misure di controllo dell'epidemia, come le vaccinazioni, si può controllare la diffusione dell'epidemia con un indicatore analogo a R_0, che descrive l'inizio del riconoscimento dell'epidemia [12]:

Definizione 1.3. Il **numero di riproduzione di controllo**, indicato con R_c, rappresenta il numero di infezioni secondarie causate da un singolo infetto

in una popolazione composta essenzialmente solo da suscettibili con le misure di controllo in atto.

Verifichiamo che, per il modello (1.19)

$$R_c = \beta_I \left(\frac{(1-\delta)\epsilon}{\gamma_a} + \frac{\delta}{\gamma} \right) \tag{1.20}$$

R_c può essere calcolato studiando gli stati di equilibrio del sistema. Il sistema (1.19) è in equilibrio quando:

$$S' = E' = I' = J' = D' = H' = R' = I^{c'} = H^{c'} = 0 \tag{1.21}$$

Per semplicità omettiamo le dipendenze temporali.

$$I' = 0 \Rightarrow I = \frac{\delta\alpha}{\gamma} E \tag{1.22}$$

$$J' = 0 \Rightarrow J = \frac{(1-\delta)\alpha}{\gamma_a} E \tag{1.23}$$

Sostituiamo (1.22) e (1.23) in $E' = 0$:

$$\frac{\beta_I (\frac{\delta\alpha}{\gamma} E + \epsilon \frac{(1-\delta)\alpha}{\gamma_a} E)}{N} S - \alpha E = 0 \tag{1.24}$$

$$\Rightarrow E \left(\left(\frac{\delta\alpha}{\gamma} + \epsilon \frac{(1-\delta)\alpha}{\gamma_a} \right) S - \frac{\alpha N}{\beta_I} \right) = 0 \tag{1.25}$$

$$\Rightarrow E^* = 0 \vee \left(\frac{\delta\alpha\gamma_a + \epsilon(1-\delta)\alpha\gamma}{\gamma_a\gamma} \right) S - \frac{\alpha N}{\beta_I} = 0 \tag{1.26}$$

$$\Rightarrow E^* = 0 \vee (\delta\alpha\gamma_a + \epsilon(1-\delta)\alpha\gamma) S = \frac{\alpha\gamma_a\gamma}{\beta_I} N \tag{1.27}$$

$$\Rightarrow E^* = 0 \vee S^* = \frac{\gamma_a\gamma}{\beta_I(\delta\gamma_a + \epsilon\gamma(1-\delta))} N \tag{1.28}$$

dove con E^* e S^* indichiamo i valori di E e S all'equilibrio.
Da (1.28) otteniamo:

$$Rc = \frac{1}{S^*} N = \beta_I \left(\frac{(1-\delta)\epsilon}{\gamma_a} + \frac{\delta}{\gamma} \right) \tag{1.29}$$

1.3 Modelli con ritardo

Spesso, la reazione di un sistema sottoposto ad un certo stimolo non è immediata ma avviene con un certo ritardo. Si usano quindi le DDEs (dall'inglese *Delay Differential Equations*) per descrivere tali fenomeni.

Definizione 1.4. La forma più generale di una DDE è:

$$y'(t) = f(t, y_t),\ t \geq t_0 \tag{1.30}$$

dove $y_t = y(t+\theta)$, $\theta \in [-r, 0]$, è una funzione dello spazio di Banach $C = C^0([-r, 0], \mathbb{R}^d)$, $f\colon \mathbb{R} \times C \supset \Omega \to \mathbb{R}^d$.

La 1.30 può essere espressa nella formulazione più semplice:

$$y'(t) = f(t, y(t-\tau_1, ..., y(t-\tau_n)),\ t \geq t_0 \tag{1.31}$$

dove $\tau_1, ..., \tau_n$ sono i *ritardi* e possono essere costanti, funzioni di t o funzioni di t e $y(t)$.

Esempio 1.1. Modello di crescita di una popolazione

Sia $N(t)$ il numero di individui di una popolazione al tempo t, $r > 0$ il tasso di crescita e $K > 0$ la *carrying capacity*, ovvero la capacità portante (il numero massimo di individui che possono trovarsi in un determinato ambiente). Con tali parametri, nel 1838 Verhulst definì la ben nota **equazione logistica** [13]:

$$\frac{dN(t)}{dt} = rN(t)\left(1 - \frac{N(t)}{K}\right) \tag{1.32}$$

Tale modello dà per scontato che se l'ambiente cambia, ad esempio la popolazione cresce e consuma più risorse, istantaneamente gli individui della specie se ne accorgono. Ma nella realtà ciò avviene dopo un certo tempo, ovvero si verifica il fenomeno del *ritardo di risposta*. Nel 1948 Hutchinson propose un modello, noto come **equazione logistica con ritardo** [14]:

$$\frac{dN(t)}{dt} = rN(t)\left(1 - \frac{N(t-\tau(t))}{K}\right) \tag{1.33}$$

La soluzione cambia in base ai diversi valori di $\tau(t)$ ed è più fedele alla realtà.

1.3.1 SIJHDR

Semplifichiamo il modello (1.19) eliminando il compartimento E e introducendo un ritardo $\tau(t)$ nella dinamica del compartimento S. Tale ritardo è

dovuto all'effetto delle vaccinazioni, visibili dopo circa 14 giorni dalle prime somministrazioni. Il diagramma di trasferimento del modello assume dunque la forma:

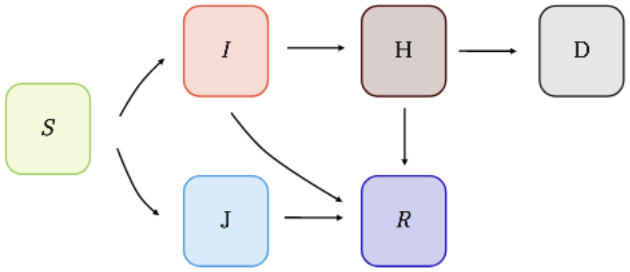

Figura 1.6: Diagramma di trasferimento per il modello (1.34)

e il modello è descritto dal sistema di ODEs:

$$\begin{cases} \frac{dS(t)}{dt} = -\frac{\beta_I(t)[I(t-\tau(t))+\epsilon J(t-\tau(t))]}{N} S(t-\tau(t)), \\ \frac{dI(t)}{dt} = \delta\frac{\beta_I(t)[I(t-\tau(t))+\epsilon J(t-\tau(t))]}{N} S(t-\tau(t)) - \gamma I(t), \\ \frac{dJ(t)}{dt} = (1-\delta)\frac{\beta_I(t)[I(t-\tau(t))+\epsilon J(t-\tau(t))]}{N} S(t-\tau(t)) - \gamma_a J(t), \\ \frac{dH(t)}{dt} = p(t)\gamma I(t) - d_H H(t), \\ \frac{dD(t)}{dt} = q(t) d_H H(t), \\ \frac{dR(t)}{dt} = \gamma_a J(t) + (1-p(t))\gamma I(t) + (1-q(t)) d_H H(t), \\ \frac{dI^c(t)}{dt} = \delta\frac{\beta_I(t)[I(t-\tau(t))+\epsilon J(t-\tau(t))]}{N} S(t-\tau(t)), \\ \frac{dH^c(t)}{dt} = p(t)\gamma I(t). \end{cases} \tag{1.34}$$

Anche per il modello (1.34), il numero di riproduzione di controllo vale:

$$R_c = \beta_I \left(\frac{(1-\delta)\epsilon}{\gamma_a} + \frac{\delta}{\gamma} \right) \tag{1.35}$$

Capitolo 2

Reti neurali artificiali

Questo secondo capitolo tratterà principalmente di reti neurali artificiali (ANNs). Dopo un'introduzione al Machine Learning, presenteremo le ANNs, la loro versione *deep* ed infine parleremo di reti neurali "Physics-Informed", concludendo con la loro applicazione al modello 1.19.
I contenuti di questo capitolo sono tratti principalmente da [15], [16], [17] e [18].

2.1 Introduzione al Machine Learning

Il **Machine Learning** (ML), tradotto in italiano con *Apprendimento Automatico*, inteso come abilità delle macchine (i computer) di apprendere senza essere state esplicitamente e preventivamente programmate, è un sottoinsieme dell'Intelligenza Artificiale (AI). La definizione più accreditata dalla comunità scientifica è quella fornita nel 1997 da Tom Michael Mitchell, informatico statunitense e professore universitario presso la Carnegie Mellon University (CMU), in [19]:

Definizione 2.1. Un programma apprende dall'esperienza E con riferimento a alcune classi di compiti T e con misurazione della performance P, se le sue performance nel compito T, come misurato da P, migliorano con l'esperienza E.

2.1.1 Processo di apprendimento

Nella pratica un'esperienza può essere tradotta in un'azione futura attraverso il seguente processo di apprendimento:

1. raccolta dati;

2. esplorazione e preparazione dei dati;
3. addestramento del modello;
4. valutazione del modello;
5. miglioramento del modello.

In base al tipo di esperienza, in particolare in base alle caratteristiche dei dati raccolti durante il processo di apprendimento, gli algoritmi di ML possono essere distinti in tre categorie:

- **apprendimento supervisionato**
  Un insieme di valori "etichettati", cioè i cui output desiderati sono noti a priori, vengono dati in input all'algoritmo. L'obiettivo è che l'algoritmo confronti il suo output effettivo con l'output desiderato e regoli il suo modello di conseguenza.

- **apprendimento non supervisionato**
  Nessun dato di input è "etichettato" ed è compito dell'algoritmo trovare aspetti in comune tra i diversi dati per classificarli di conseguenza.
  É particolarmente utilizzato per problemi di *clustering*: lo scopo è quello di raggruppare una collezione di oggetti "simili" in sottoinsiemi o cluster, in modo tale che quelli all'interno di ogni cluster siano più strettamente correlati tra loro rispetto agli oggetti assegnati a cluster diversi.
  Nei modelli di apprendimento non supervisionato, la mancanza di etichette sui dati di addestramento rende problematica la valutazione del modello, in quanto non c'è nulla di significativo con cui i risultati possono essere confrontati ed è necessario definire delle metriche alternative.

- **apprendimento per rinforzo**
  I dati di input non sono un set di dati fisso. C'è un ciclo di feedback tra il sistema di apprendimento e le sue esperienze. In particolare, se dopo una sequenza di azioni il feedback è positivo la macchina riceve una **ricompensa**, altrimenti riceve una **penalizzazione**. L'esperienza viene memorizzata e il sistema di apprendimento cercherà di non ripetere le azioni che hanno generato una perdita.

Esempio 2.1. Classificazione
Un classico problema in cui si utilizzano metodi di ML è quello di classificazione. In particolare, tale problema può essere formulato secondo la definizione di Mitchell specificando:

Esperienza E: insieme di immagini di k animali, tradotte in un dataset composto da coppie $(\mathbf{x}, y)$, dove $\mathbf{x} \in \mathbb{R}^n$ è l'insieme di valori di luminosità dei pixel di un'immagine e $y \in \{1, ..., k\}$ è l'etichetta che indica la classe di appartenenza dell'animale presente nell'immagine (es. 1 per il gatto, 2 per il cane ecc..).

Compito T : data una nuova immagine contenente un animale si chiede di classificarla, ovvero dato $\mathbf{x}$, trovare la y corrispondente. La previsione di y, dato $\mathbf{x}$, viene di solito effettuata andando a massimizzare la probabilità condizionata definita come:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}, y)}{\sum_{y'} p(\mathbf{x}, y')} \tag{2.1}$$

Performance P: accuracy (precisione). É la proporzione di esempi per i quali il modello produce l'output corretto. Viene valutata comparando i risultati ottenuti su un test set (un insieme di dati separato dai dati utilizzati per l'addestramento del sistema di apprendimento automatico), con le etichette presenti nel test set stesso.

Il processo di apprendimento può avvenire su più *layers*, livelli in cui il programma dovrà risolvere compiti via via più difficili. In tal caso si parla di **Deep Learning** (DL), tradotto letteralmente in *Apprendimento Profondo*. In 2.1 vengono mostrate le inclusioni di DL in ML, a sua volta nell'AI.

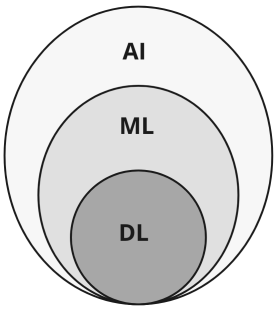

Figura 2.1: Diagramma di Venn che mostra la relazione tra AI, ML e DL

L'approccio del DL è essenzialmente basato sulle **reti neurali artificiali**, modelli computazionali ispirati al funzionamento dei neuroni nel cervello umano.

2.2 Il Percettrone

Il **percettrone** è il più semplice modello di NN. È stato proposto dallo psicologo americano Frank Rosenblatt nel 1957 [20] e si ispira al neurone biologico, schematizzato in 2.2.

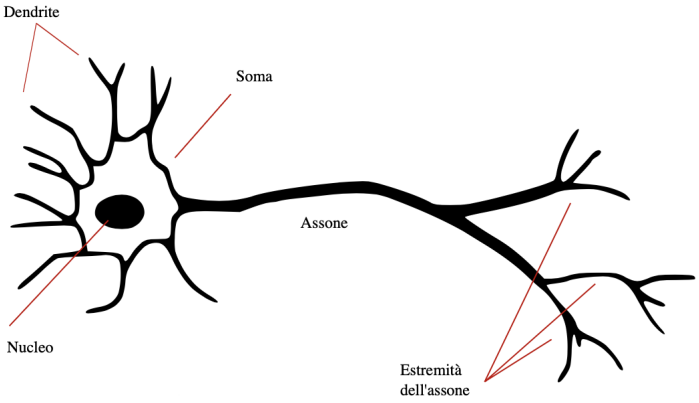

Figura 2.2: Diagramma di un neurone
Fonte:[21]

Il neurone biologico riceve in ingresso segnali da altri neuroni attraverso i dendriti. Tali segnali vengono integrati dal soma (il corpo del neurone) e, se l'attivazione che ne risulta supera una certa soglia s, viene generato un potenziale d'azione che si propaga attraverso il suo assone ad uno o più neuroni. Siano:

- x_1, x_2,..., x_n i segnali di input binari;
- w_1, w_2,..., w_n i pesi, numeri reali che esprimono l'importanza dei rispettivi input per l'output;
- f la funzione di *attivazione*, detta anche funzione *primitiva*; per il percettrone è la funzione a gradino unitario di Heaviside 2.2;

$$f(x) = \begin{cases} 0, x < 0 \\ 1, x \geq 0 \end{cases} \tag{2.2}$$

- y l'output binario;
- b il bias, una misura di quanto sia facile far sì che il percettrone emetta un 1. Corrisponde all'opposto del valore di soglia del neurone biologico, ovvero $b = -s$.

Il processo biologico viene semplificato in 2.3.

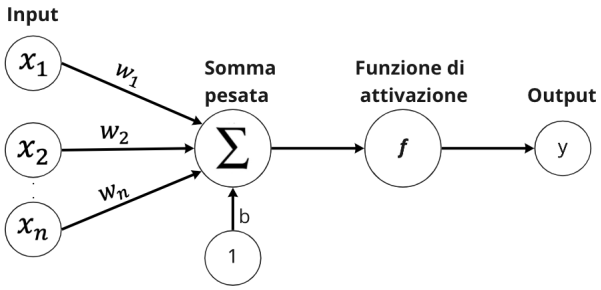

Figura 2.3: Schema del percettrone

.

Al soma giunge la somma pesata:

$$\sum = x_1 w_1 + x_2 w_2 + ... + x_n w_n + b \tag{2.3}$$

in cui verrà valutata la funzione primitiva, $f(\sum) = y$. In particolare:

1. *se* $\sum < 0 \Rightarrow y = 0$
2. se $\sum \geq 0 \Rightarrow y = 1$

Osservazione 5. L'insieme di punti per cui

$$\sum = 0 \tag{2.4}$$

definisce un iperpiano nello spazio delle variabili x_i, $i = 1, .., n$. Esso divide lo spazio nelle due parti 1 e 2.

Il percettrone 2.3 è il più semplice esempio di rete neurale *feedforward* (FFNN), in cui l'informazione è unidirezionale, dallo strato di Input allo strato di output. Il percettrone è particolarmente utilizzato nei problemi di classificazione binaria.

2.3 Reti neurali profonde (DNNs)

Per studiare problemi più complessi, negli anni ottanta sono state introdotte le reti neurali DNNs, anche note come reti neurali multistrato. Esse, infatti, funzionano allo stesso modo delle reti neurali classiche, con la differenza che hanno un numero molto elevato di neuroni contenuti nei livelli intermedi, anche detti livelli nascosti o strati latenti. Analizziamo la struttura delle DNNs, mostrata in figura 2.4, da sinistra verso destra:

- il primo livello, detto strato di input o *input layer*, contiene i neuroni di input;
- l'ultimo livello è chiamato strato di output o *output layer* e contiene neuroni di output;
- i livelli centrali, chiamati strati latenti o *hidden layers*, contengono neuroni nè di input nè di output.

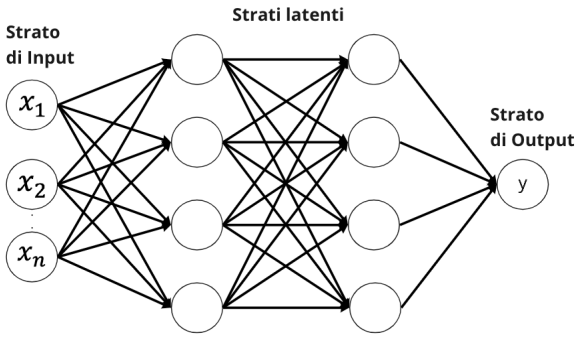

Figura 2.4: Schema di una DNN *feedforward* con due strati latenti

.

Considereremo DNNs di tipo *feedforward* (FFDNNs), ovvero DNN unidirezionali, in cui l'output di uno strato è usato come input per lo strato successivo.

Funzione di costo

Per determinare il mapping desiderato tra i valori presenti nello strato di Input $\mathbf{x}=\{x_1, .., x_N\}$ e quelli nello strato di Output $\mathbf{y}=\{y(x_1), ..., y(x_N)\}$ è necessario trovare i valori ottimali dei pesi. Tale ricerca avviene durante la fase di addestramento, attraverso la attraverso la minimizzazione di una funzione di costo (o di perdita), comunemente nota come *loss function*:

$$L = L(\mathbf{x}, \mathbf{w}) \tag{2.5}$$

che dipende dai dati $\mathbf{x}$ e dai parametri (i pesi) $\mathbf{w}$.
Consideriamo una rete composta da L strati e denotiamo, per $l = 1, 2, .., L$:

- n_l il numero di neuroni dello strato l;
- $W^{[l]} \in \mathbb{R}^{n_l \times n_{l-1}}$ la matrice dei pesi allo strato l; in particolare $w_{jk}^{[l]}$ è il peso che il neurone j allo strato l applica all'output dal neurone k allo strato $l-1$.

- $b^{[l]} \in \mathbb{R}^{n_l}$ il vettore dei *biases* per lo strato l;
- $a^{[l]}$ l'output dello strato l;
- $z^{[l]} = b^{[l]} + W^{[l]}a^{[l-1]}$ il vettore di input pesato per lo strato l;
- f la funzione di attivazione, uguale per tutti gli strati.

. In particolare $a^{[1]} = \mathbf{x}$, $a^{[L]} = \mathbf{y}$ e $a^{[l]} = f(z^{[l]})$, $l = 2, .., L-1$.

Una funzione costo molto usata è la *funzione di costo quadratica* (MSE):

$$L(\mathbf{w}) = MSE = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}||y(x_i) - a^{[L]}(x_i)||_2^2 \tag{2.6}$$

dove $\mathbf{w} = (W^{[1]}, b^{[1]}, \ldots, W^{[L]}, b^{[L]})$.

2.3.1 Retropropagazione

L'algoritmo di addestramento di reti neurali più studiato e utilizzato è *backpropagation* [22], o *retropropagazione* (BP) ed è usato in combinazione con un metodo di ottimizzazione, come per esempio la **discesa del gradiente**. Attraverso la BP vengono calcolate le derivate della funzione di perdita rispetto ai pesi, ovvero il gradiente di L "pensata" come funzione di $\mathbf{w}$:

$$\nabla_{\mathbf{w}} L(\mathbf{w}) = \left[\frac{\partial L(\mathbf{w})}{\partial w_1}, \frac{\partial L(\mathbf{w})}{\partial w_2}, ..., \frac{\partial L(\mathbf{w})}{\partial w_n}\right] \tag{2.7}$$

Il nome *retroprapagazione* si riferisce al fatto che il gradiente è ottenuto dalla *propagazione all'indietro* del valore della funzione obiettivo per i pesi $\mathbf{w}$, utilizzando la *chain rule*, ovvero la regola della catena, per calcolare le derivate parziali della funzione obiettivo rispetto ad ogni peso.

La BP si sviluppa principalmente in 2 passi [23]:

1. forward pass
2. backward pass

Forward pass

La prima azione da compiere nel forward pass è inizializzare i pesi della rete. Di solito ciò avviene in modo casuale. Viene quindi effettuato un primo giro (*epoch*) di addestramento della rete e, per ogni strato, vengono salvate le derivate parziali rispetto a $z^{[l]}$, $l = L, .., 1$.

Backward pass
Utilizziamo la notazione vettoriale, diamo quindi la definizione di prodotto di Hadamard [24]:

Definizione 2.2. Si definisce prodotto di Hadamard tra due matrici con lo stesso numero di righe e colonne, la matrice in cui ogni elemento è il prodotto degli elementi corrispondenti nelle due matrici.
Si indica con $\odot$ e, se x e y sono due vettori n-dimensionali, il loro prodotto di Hadamard è un vettore in cui ogni elemento è il prodotto tra le componenti dei vettori corrispondenti, ovvero:

$$(x \odot y)_j = x_j y_j \tag{2.8}$$

Per $j \in [1, n_l]$ e $l \in [2, L]$ poniamo:

$$\delta_j^{[l]} = \frac{\partial L}{\partial z_j^{[l]}} \tag{2.9}$$

Si ottengono dunque i seguenti risultati:

Teorema 2.1.

$$\begin{aligned} \delta^L &= f'(z^{[L]}) \odot \frac{\partial L}{\partial z^{[L]}} \\ \delta^l &= f'(z^{[l]}) \odot (W^{[l+1]})^T \delta^{[l+1]} \\ \frac{\partial L}{\partial b_j^{[l]}} &= \delta_j^{[l]} \\ \frac{\partial L}{\partial w_{jk}^{[l]}} &= \delta_j^{[l]} a_k^{[l-1]} \end{aligned} \tag{2.10}$$

Per la dimostrazione, ottenuta applicando più volte la regola della catena, rimandiamo a [23].
Si possono quindi aggiornare i pesi e i *biases* e ricominciare: per $l = 1, 2, .., L$,

$$\begin{aligned} W^{[l]} &= W^{[l]} - \mu \delta_j^{[l]} a^{[l-1]^T} \\ b^{[l]} &= b^{[l]} - \mu \delta_j^l \end{aligned} \tag{2.11}$$

dove μ indica il *learning rate*, tasso di apprendimento e determina la dimensione dello spostamento compiuto per raggiungere il punto di minimo (locale) nella discesa del gradiente.

2.3.2 La differenziazione automatica

Molti algoritmi di ML richiedono il calcolo delle derivate di una funzione e ci sono quattro metodi per farlo:

1. elaborazione **manuale** delle derivate e loro codifica;

2. differenziazione **numerica** usando approssimazioni alle differenze finite (FDM);

3. differenziazione **simbolica** usando la manipolazione delle espressioni nei sistemi di algebra del computer;

4. differenziazione **automatica**, chiamata anche differenziazione algoritmica (AD).

L'AD è una controparte specializzata della BP.
L'AD [25] esegue un'interpretazione non standard di un dato programma informatico sostituendo il dominio delle variabili per incorporare valori di derivazione e ridefinendo la semantica degli operatori per propagare le derivate secondo la regola della catena del calcolo differenziale. L'AD lavora dunque sull'implementazione di un'espressione simbolica in qualche linguaggio di programmazione.

Ne introduciamo le maggiori caratteristiche attraverso un esempio.

Esempio 2.2. Calcolo delle derivate parziali di una funzione usando la differenziazione automatica. Sia

$$f(w_1, w_2) = w_2 \cdot sen(2w_2 + w_1) \tag{2.12}$$

Per trovare $\frac{\partial f(w_1,w_2)}{\partial w_2}$, l'idea è di applicare le regole delle derivate alle singole operazioni elementari, ovvero a:

$$\begin{cases} w_1 \\ w_2 \\ z_1 = 2w_2 \\ z_2 = z_1 + w_1 \\ z_3 = sen(z_2) \\ z_4 = w_2 z_3 \end{cases} \tag{2.13}$$

Esprimiamo in maniera visiva i passi intermedi di 2.12 nella figura 2.5.

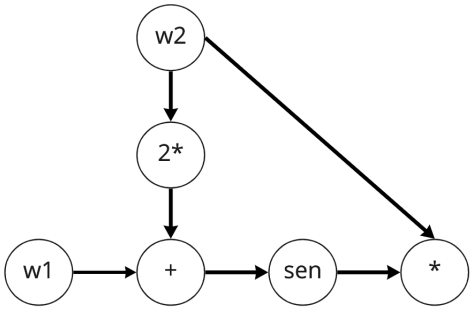

Figura 2.5: Grafo computazionale dell'espressione iniziale.

.

Algoritmo di AD *forward-mode*

Ad ogni nodo del grafo associamo un secondo valore, detto **numero duale**, che rappresenti la derivata numerica di quel nodo rispetto a w_2 applicando, dove necessario, la regola della catena. In questo modo i valori che andiamo a calcolare dipendono solo dalla derivata elementare del singolo nodo e dai valori calcolati nei nodi precedenti sul grafo.
Ponendo $dw_j = \frac{\partial w_j}{\partial w_2}$, $j = 1, 2$ e $dz_i = \frac{\partial z_i}{\partial w_2}$, $i = 1, 2, 3, 4$ si ottiene:

$$\begin{cases} dw_1 = 0 \\ dw_2 = 1 \\ dz_1 = 2 \\ dz_2 = 2 = dz_1 + dw_1 \\ dz_3 = 2cos(z_2)) = dz_1 * \frac{\partial sen(z_2)}{\partial z_2} \\ dz_4 = z_3 + 2cos(z_2) = z_3 + dz_3 \end{cases} \tag{2.14}$$

Osserviamo che nell'implentazione numerica dell'algoritmo visto, si devono inizializzare due valori: $dw_1 = 0$ e $dw_2 = 1$.

Algoritmo di AD *reverse-mode*

Una variante della AD *forward-mode* è la AD *reverse-mode*, basata sulla percorrenza *al contrario* del grafo computazionale di un'espressione. Esso corrisponde ad un algoritmo di backpropagation generalizzato, in quanto propaga le derivate all'indietro da un dato output. Costruiamo un nuovo insieme di variabili intermedie, che descriveranno la derivata parziale di tutti i nodi del grafo rispetto all'output finale.
Nell'esempio, ciò equivale a calcolare $gz_i = \frac{\partial z_i}{\partial z_4}$, i=4,3,2,1 e $gw_j = \frac{\partial w_j}{\partial z_4}$, j=2,1.

Otteniamo:

$$\begin{cases} gz_4 = 1 \\ gw_2 = \frac{1}{z_3}, gz_3 = \frac{1}{w_2} \\ \Rightarrow \frac{1}{w_2} = cos(z_2) gz_2 \\ \Rightarrow gz_2 = \frac{1}{w_2 * cos(z_2)} \\ gz_1 = 2gw_2 = \frac{2}{z_3} \\ \frac{1}{w_2 * cos(z_2)} = gz_1 + gw_1 = \frac{2}{z_3} + gw_1 \\ \Rightarrow gw_1 = \frac{1}{w_2 * cos(z_2)} - \frac{2}{z_3} \end{cases} \tag{2.15}$$

Nel reverse-mode si possono quindi calcolare tutte le derivate parziali degli input rispetto ad un singolo output con **una sola iterazione**. In generale, il numero di parametri di una rete neurale è molto grande e l'output (la funzione costo) è unico quindi il *reverse-mode* risulta più efficiente del *forward-mode* e degli altri algoritmi per il calcolo delle derivate.

2.4 Reti neurali "physics-informed" (PINNs)

Le reti neurali *physics-informed*, letteralmente *informate sulla fisica*, sono state introdotte per la prima volta da Raissi ed altri [3] per risolvere problemi inversi (determinare i parametri di un modello a partire da dati noti) e in avanti (predire l'evoluzione di un sistema a partire dai parametri osservati) che coinvolgono equazioni differenziali alle derivate parziali non lineari (NPDEs). Le PINNs sono addestrate a risolvere compiti di apprendimento supervisionato rispettando leggi fisiche descritte da NPDEs.

Per differenziare le reti neurali rispetto alle loro coordinate di input e ai parametri del modello per ottenere PINNs si fa uso della AD.

Consideriamo una generale PDE:

$$u_t(t,x) = \mathcal{F}(u(t,x)), x \in \Omega \subseteq \mathbb{R}^D, t \in [0,T] \tag{2.16}$$

dove $u(t,x)$ denota la soluzione cercata, u_t la sua derivata rispetto alla variabile t e $\mathcal{F}(\cdot)$ un operatore differenziale non lineare.
Sia

$$\mathcal{R}(t,x) := u_t(t,x) - \mathcal{F}(u(t,x)) \tag{2.17}$$

il residuo della PDE. Mostriamo, con un esempio, come è possibile *approssimare* la soluzione $u(t,x)$ con una DNN.

Esempio 2.3. Equazione di Burger
Esaminiamo l'equazione di Burger in uno spazio unidimensionale, con condizioni al contorno (BC) di Dirichlet:

$$\begin{cases} u_t + uu_x - \left(\frac{0.01}{\pi}\right) u_{xx} = 0, x \in [-1,1], t \in [0,1] \\ u(0,x) = -sin(\pi x) \\ u(t,-1) = u(t,1) = 0 \end{cases} \tag{2.18}$$

Calcoliamo le derivate u_t, u_x, u_{xx} con la AD. L'algoritmo di AD *reverse-mode* è stato implementato nel modulo

$$tensorflow.gradients() \tag{2.19}$$

di Tensorflow [26]. In esso, infatti, specificando le operazioni in un grafo computazionale, Tensorflow esegue automaticamente la regola della catena attraverso il grafico e, poiché conosce le derivate di ogni singola operazione, può combinarli automaticamente.
Supposto che `neural_net` sia una funzione in cui è definita l'architettura della rete neurale, definiamo u nel seguente codice Python:

```
def u(t, x):
    u = neural_net(tf.concat([t,x],1), weights, biases)
    return u
```

L'AD permette di approssimare u con $\mathcal{R}$ attraverso poche linee di codice:

```
def R(t, x):
    u = u(t, x)
    u_t = tf.gradients(u, t)[0]
    u_x = tf.gradients(u, x)[0]
    u_xx = tf.gradients(u_x, x)[0]
    R = u_t + u*u_x - (0.01/tf.pi)*u_xx
    return R
```

I parametri possono infine essere appresi minimizzando la funzione di perdita, definita come la somma di due errori quadratici medi (MSE):

$$\begin{aligned} L(\mathbf{w}) =& MSE_u + MSE_r = \\ =& \frac{1}{N_u} \sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2 + \frac{1}{N_r} \sum_{i=1}^{N_r} |\mathcal{R}(t_r^i, x_r^i)|^2 \end{aligned} \tag{2.20}$$

dove $(t_u^i, x_u^i, u^i)_{i=1}^{N_u}$ denotano rispettivamente i punti nel dominio spazio/tempo e il valore della funzione $u(t,x)$ osservato in quei punti. In particolare sono N_u punti relativi al dato al bordo e ai dati che si hanno a disposizione.

Mentre $(t_r^i, x_r^i)_{i=1}^{N_r}$ sono gli Nr punti residui (detti anche di collocazione), ovvero punti in cui forziamo l'ODE ad essere soddisfatta. Solitamente vengono selezionati casualmente su tutto il dominio di integrazione.

In particolare:

- MSE_u pesa quanto la soluzione differisce dai dati nei punti di addestramento;
- MSE_r penalizza l'equazione che governa il modello nei punti di collocazione in modo che rispetti il vincolo imposto dall'equazione differenziale.

2.4.1 Applicazione delle PINNs al modello SEIJDHR

Durante il corso della malattia COVID-19 sono molti i fattori che variano a causa, principalmente, di mutazioni dell'agente patogeno e del cambiamento dei sintomi. É importante, dunque, considerare parametri dipendenti dal tempo, come visto nel modello matematico (1.19). In questo caso la PINN è utilizzata con duplice scopo:

1. dedurre tali parametri;
2. adattare i dati per fare previsioni.

Struttura della rete

Sia $U(t) = [S(t), E(t), I(t), J(t), D(t), H(t), R(t), I^c(t), H^c(t)]$ la soluzione del sistema (1.19). U viene approssimata attraverso una rete neurale U_{NN} con le seguenti caratteristiche:

- 10 strati latenti;
- 32 neuroni per strato;
- *tanh* come funzione di attivazione.

I parametri $\beta_I(t)$, $p(t)$, $q(t)$ e $d(t)$ (l'ultimo relativo al modello (1.34)) vengono dedotti da altre 4 reti che presentano la seguente struttura:

- 5 strati latenti;
- 20 neuroni per strato;
- *tanh* come funzione di attivazione.

La struttura della PINN è schematizzata in figura 2.6.

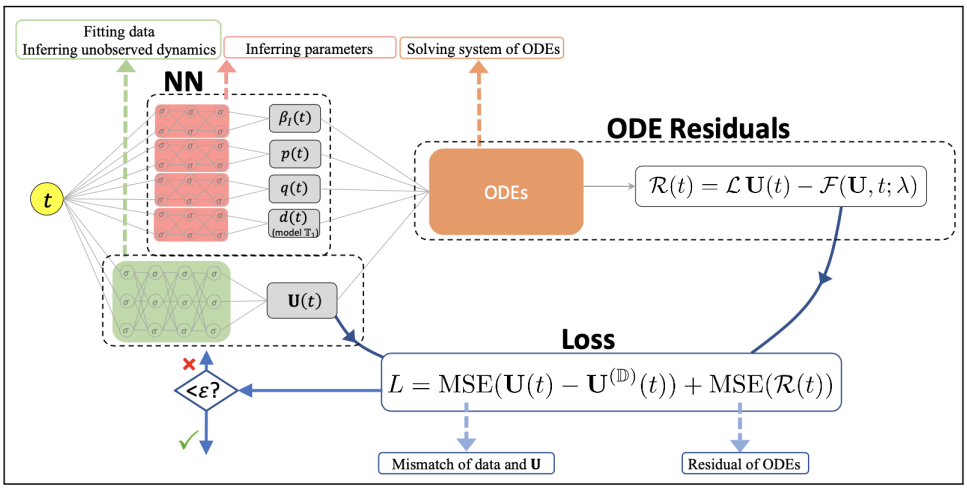

Figura 2.6: Schema delle PINNs relative ai modelli (1.19) e (1.34)
NN denota diverse reti neurali che rappresentano gli stati $U(t)$ (area ombreggiata in verde) e i parametri $\beta_I(t)$, $p(t)$, $q(t)$ nel modello (1.19) e $d(t)$ nel modello (1.34) (area ombreggiata in rosso).
Nel riquadro **ODE Residuals** viene calcolato il residuo $\mathcal{R}(t)$ dei modelli.
Nel riquadro **Loss** viene definita la funzione di perdita L, composta da un termine di mismatch tra dati e NN e uno del residuo di ODEs.
Fonte: Informazioni supplementari di [4]

Funzione di costo

Siano $U_{NN}(t; \mathbf{w})$ la DNN che approssima $U(t)$, con input t, parametrizzata da $\mathbf{w}$ (pesi e *biases* della rete); $\mathcal{F}(U_{NN}, t; \lambda)$ il vettore avente per componenti i secondi membri del sistema di ODE 1.19, codificate nella rete, con parametri λ e $\mathcal{R}_{NN}(t) := \frac{d}{dt}U_{NN}(t) - \mathcal{F}(U_{NN}, t; \lambda)$ il residuo.

La funzione di perdita è definita come la somma di due termini:

$$\begin{aligned} L(\mathbf{w}, \lambda) =& \omega_u MSE_u + \omega_r MSE_r = \\ =& \omega_u \frac{1}{N_u} \sum_{j=1}^{N_u} |U_{NN}(t_u^j) - U^{Data}(t_u^j)|^2 + \omega_r \frac{1}{N_r} \sum_{j=1}^{N_r} |\mathcal{R}_{NN}(t_r^j)|^2 \end{aligned} \tag{2.21}$$

Osservazione 6. Il valore di MSE_u dipende dalla disponibilità di dati sulle classi epidemiologiche. MSE_r dipende dal valore dell'equazione nei punti residui (punti in cui il residuo $\mathcal{R}_{NN}(t)$ è soddisfatto).

Supponiamo che le classi epidemiologiche disponibili sono $I^c(t), D^c(t), H^c(t)$. Da esse si ricavano facilmente i *nuovi* valori giornalieri:

$$\begin{aligned} I^{new}(t) &= I^c(t) - I^c(t-1) \\ H^{new}(t) &= H^c(t) - H^c(t-1) \\ D^{new}(t) &= D^c(t) - D^c(t-1) \end{aligned} \tag{2.22}$$

Calcoliamo, in tal caso, $N_u * MSE_u$:

$$\begin{aligned} &\sum_{j=1}^{N_u} |I^{new}_{NN}(t_j) - I^{new}(t_j)|^2 + \sum_{j=1}^{N_u} |H^{new}_{NN}(t_j) - H^{new}(t_j)|^2 + \sum_{j=1}^{N_u} |D^{new}_{NN}(t_j) - D^{new}(t_j)|^2 + \\ &+ \sum_{j=1}^{N_u} |I^c_{NN}(t_j) - I^c(t_j)|^2 + \sum_{j=1}^{N_u} |H^c_{NN}(t_j) - H^c(t_j)|^2 + \sum_{j=1}^{N_u} |D^c_{NN}(t_j) - D^c(t_j)|^2 \end{aligned} \tag{2.23}$$

Vediamo la forma assunta da $N_r * MSE_r$:

$$\begin{aligned} &\sum_{j=1}^{N_r} \left| \frac{d}{dt} E_{NN}(t_j) - \frac{\beta_{INN}(t_j)[I(t_j) + \epsilon J(t_j)]}{N} S_{NN}(t_j) + \alpha E_{NN}(t_j) \right|^2 + \\ &+ \sum_{j=1}^{N_r} \left| \frac{d}{dt} I_{NN}(t_j) - \delta\alpha E_{NN}(t_j) + \gamma I_{NN}(t_j) \right|^2 + \sum_{j=1}^{N_r} \left| \frac{d}{dt} D_{NN}(t_j) - q_{NN}(t_j)\Phi_D H_{NN}(t_j) \right|^2 + \\ &+ \sum_{j=1}^{N_r} \left| \frac{d}{dt} J_{NN}(t_j) - (1-\delta)\alpha E_{NN}(t_j) + \gamma_a J_{NN}(t_j) \right|^2 + \\ &+ \sum_{j=1}^{N_r} \left| \frac{d}{dt} H_{NN}(t_j) - p_{NN}(t_j)\gamma I_{NN}(t_j) + q_{NN}(t_j)\Phi_D H_{NN}(t_j) + (1 - q_{NN}(t_j))\Phi_R H_{NN}(t_j) \right|^2 + \\ &+ \sum_{j=1}^{N_r} \left| \frac{d}{dt} R_{NN}(t_j) - \gamma_a J_{NN}(t_j) - (1 - p_{NN}(t_j))\gamma I_{NN}(t_j) - (1 - q_{NN}(t_j))\Phi_R H_{NN}(t_j) \right|^2 + \\ &+ \sum_{j=1}^{N_r} \left| \frac{d}{dt} I^c_{NN}(t_j) - \delta\alpha E_{NN}(t_j) \right|^2 + \sum_{j=1}^{N_r} \left| \frac{d}{dt} H^c_{NN}(t_j) - p_{NN}(t_j)\gamma I_{NN}(t_j) \right|^2 \end{aligned} \tag{2.24}$$

con $S_{NN}(t) = N - E_{NN}(t) - I_{NN}(t) - J_{NN}(t) - D_{NN}(t) - H_{NN}(t) - R_{NN}(t)$.

Capitolo 3

Dati COVID-19

In questo capitolo vengono introdotti gli strumenti utilizzati per leggere ed estrarre i dati e viene poi data una panoramica sui dati disponibili sulla malattia COVID-19, accessibili a tutti e non, con particolare riguardo ai dati italiani e newyorkesi.

Dalla scoperta del primo caso di COVID-19 in Cina nel dicembre del 2019, sono stati creati numerosi archivi di dati, che hanno consentito agli studiosi di avere una visione più ampia sullo sviluppo della malattia. Essi hanno effettuato numerosi studi e pubblicato più di 100000 articoli.

Al fine di comprendere al meglio l'evoluzione di una pandemia è necessario 'leggere' correttamente i dati. In questo lavoro i dati sono stati letti utilizzando **Jupyter Notebook (JN)**: un'applicazione web open source che consente di includere testo, video, audio e immagini e offre la possibilità di eseguire codice in diversi linguaggi di programmazione. Il linguaggio di programmazione scelto è Python e si è fatto ampio uso delle librerie:

- *pandas* per caricare ed analizzare i dati;
- *matplotlib.pyplot* per creare visualizzazioni statiche, animate o interattive.

Nell'aprile 2008 Tom Preston-Werner, Chris Wanstrath, P. J. Hyett e Scott Chacon hanno lanciato il sito `https://github.com`, acquisito poi da Microsoft nel 2018. Github è una piattaforma che ospita progetti software ed è basata sul software open source Git [27]. Uno degli strumenti più utilizzati dagli sviluppatori sono i **repositories github**: spazi di archiviazione strutturati che contengono del codice, salvato dagli utenti nelle versioni *master* (di produzione), *develop* (di sviluppo) ed altre.

A gennaio 2022 sono più di 190000 i repositories github riguardanti la malattia COVID-19. In questo lavoro di tesi sono stati analizzati principalmente dati caricati da cinque repositories, che vediamo di seguito.

3.1 John Hopkins University

Le principali fonti di informazione mondiale su dati riguardanti la malattia COVID-19 sono i repositories github forniti dai seguenti centri della John Hopkins University (JHU):

1. *Center for Systems Science and Engineering* (CSSE) [28]
2. *Centers for Civil Impact* (CCI) [29]

3.1.1 CSSE

In [28] troviamo la cartella 'csse_covid_19_data', aggiornata giornalmente. Essa contiene, tra le altre, la cartella 'csse_covid_19_time_series' su aggiornamenti giornalieri, mondiali e degli Stati Uniti, riguardanti:

- casi confermati di Covid-19: 'time_series_covid19_confirmed_global.csv', 'time_series_covid19_confirmed_US.csv'
- numero di deceduti: 'time_series_covid19_deaths_global.csv', 'time_series_covid19_deaths_US.csv'
- numero di guariti: 'time_series_covid19_recovered_global.csv'

Tali file contengono:
- quattro colonne relative alla provincia/stato, paese/regione, latitudine e longitudine (**Province/State**, **Country/Region**, **lat**, **long**);
- numerose colonne relative alla data, una al giorno a partire dall'1/22/20 (formato della data: mm/gg/aa).
Il numero di righe, invece, è determinato dagli stati e/o le relative regioni che forniscono i dati e seguono un indice numerico, in correlazione con l'ordine alfabetico dei valori nella colonna **Country/Region**.

Esempio 3.1. Eseguendo il seguente codice su JN

```
import pandas as pd
url_confirmed_global="https://raw.githubusercontent.com/
CSSEGISandData/COVID-19/master/csse_covid_19_data/
```

```
csse_covid_19_time_series/
    time_series_covid19_confirmed_global.csv"
df_c_g=pd.read_csv(url_confirmed_global)
df_c_g.head(1)
```

otteniamo:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	2/4/22	2/5/22	2/6/22	2/7/22
0	NaN	Afghanistan	33.939110	67.709953	0	0	0	0	0	0	...	165358	165711	166191	166924

Figura 3.1: Prima riga del file 'time_series_covid19_confirmed_global.csv'

Osservazione 7. Notiamo che non è necessario scaricare i file da leggere ed analizzare ma è sufficiente inserire il corretto *url* di github.

Se si è interessati all'andamento temporale del numero di casi confermati, si può procedere con delle trasformazioni del dataset, utilizzando, ad esempio, la funzione *pandas.melt* [30].

3.1.2 CCI

In questo archivio troviamo diversi tipi di dati tra cui quelli sui vaccini anti COVID-19 a livello mondiale: `https://github.com/govex/COVID-19/tree/master/data_tables/vaccine_data/global_data`.

Mostriamo più nel dettaglio uno dei file nella cartella precedente: 'time_series_covid19_vaccine_global.csv'. In esso ogni riga è definita in modo univoco per Paese e data; le 8 colonne di cui è composto sono descritte in tabella 3.1.

Esempio 3.2. Codice per estrarre dati sui vaccini in Italia. L'output è mostrato in figura 3.2.

```
import pandas as pd
url_time_series_vaccine_global="https://raw.githubuserco
ntent.com/govex/COVID-19/master/data_tables/vaccine_data/
global_data/time_series_covid19_vaccine_global.csv"
df_ts_vg=pd.read_csv(url_time_series_vaccine_global)
vac_ita_ts=df_ts_vg[df_ts_vg['Country_Region']=='Italy']
vac_ita_ts
```

Osservazione 8. Nella figura 3.2 osserviamo che la data contenuta nella prima riga è il 27 gennaio 2020, il giorno in cui in Italia è iniziata la somministrazione delle prime dosi di vaccini. Inoltre i dati iniziali sono nazionali, mentre nelle ultime righe la granularità è regionale. Ciò è dovuto al fatto che, i dati sui vaccini forniti dal Ministero della Salute italiano al CCI attraverso il report [31], sono aggregati per regioni a partire dall'11 maggio 2021.

NOME DELLA METRICA	DESCRIZIONE
Country_Region	Nome del Paese o della regione
Date	Data di raccolta dei dati formato aaaa-mm-gg
Doses_admin	Numero cumulativo di dosi somministrate. Quando un vaccino richiede più dosi, ognuna viene contata indipendentemente.
People_partially_vaccinated	Numero cumulativo di persone che hanno ricevuto almeno una dose di vaccino
People_fully_vaccinated	Numero cumulativo di persone che hanno ricevuto tutte le dosi necessarie per essere considerate completamente vaccinate
Report_Date_String	Data di riferimento dei dati
UID	Codice paese
Province_State	Nome della provincia o dello stato

Tabella 3.1: Descrizione del contenuto delle colonne del dataset 'time_series_covid19_vaccine_global.csv'

	Country_Region	Date	Doses_admin	People_partially_vaccinated	People_fully_vaccinated	Report_Date_String	UID	Province_State
99	Italy	2020-12-27	7177.0	7177.0	0.0	2020-12-27	380.0	NaN
128	Italy	2020-12-28	8602.0	8602.0	0.0	2020-12-28	380.0	NaN
163	Italy	2020-12-29	9611.0	9611.0	0.0	2020-12-29	380.0	NaN
200	Italy	2020-12-30	14339.0	14339.0	0.0	2020-12-30	380.0	NaN
241	Italy	2020-12-31	39821.0	39821.0	0.0	2020-12-31	380.0	NaN
...	...	...	...	...	...	...	...	...
170789	Italy	2022-02-15	9967212.0	NaN	NaN	2022-02-16	38019.0	Sicilia
170790	Italy	2022-02-15	8443243.0	NaN	NaN	2022-02-16	38009.0	Toscana
170791	Italy	2022-02-15	1975751.0	NaN	NaN	2022-02-16	38010.0	Umbria
170792	Italy	2022-02-15	268022.0	NaN	NaN	2022-02-16	38002.0	Valle d'Aosta
170793	Italy	2022-02-15	10690951.0	NaN	NaN	2022-02-16	38005.0	Veneto

Figura 3.2: Contenuto del file 'time_series_covid19_vaccine_global.csv' dal 27/12/2020 al 15/02/2022 mostrato con un Pandas DataFrame

3.2 Dati NYC

Per applicare le PINNs al modello 1.19 si è fatto uso dei dati newyorkesi riguardanti il numero di nuovi casi/ospedalizzati/deceduti giornalieri. In particolare è stato usato l'archivio dati [32]. Nella cartella 'trends' troviamo

il file 'data-by-day.csv', aggiornato giornalmente. Esso è composto da 26 colonne, di cui quattro schematizzate in tabella 3.2.

NOME DELLA METRICA	DESCRIZIONE
date_of_interest	data di riferimento
CASE_COUNT	numero di nuovi casi positivi
HOSPITALIZED_COUNT	numero di nuovi individui ospedalizzati
DEATH_COUNT	numero di nuovi individui deceduti

Tabella 3.2: Descrizione di quattro campi del database newyorkese [32]

Mostriamo il contenuto di quattro colonne di tale file eseguendo il seguente script Python:

```
import pandas as pd
url="https://raw.githubusercontent.com/nychealth/coronavirus-
    data/master/trends/data-by-day.csv"
df_nyc=pd.read_csv(url)
df_nyc.loc[:,["date_of_interest","CASE_COUNT","
    HOSPITALIZED_COUNT", "DEATH_COUNT"]]
```

	date_of_interest	CASE_COUNT	HOSPITALIZED_COUNT	DEATH_COUNT
0	02/29/2020	1	1	0
1	03/01/2020	0	1	0
2	03/02/2020	0	2	0
3	03/03/2020	1	7	0
4	03/04/2020	5	2	0
...	...	...	...	...
717	02/15/2022	892	54	18
718	02/16/2022	888	56	19
719	02/17/2022	734	43	21
720	02/18/2022	594	16	12
721	02/19/2022	358	6	5

Figura 3.3: Visualizzazione di quattro colonne del file 'data-by-day.csv'

3.3 Dati italiani

Il 31 gennaio 2020, il Consiglio dei Ministri italiano ha dichiarato lo stato di emergenza, per la durata di sei mesi, in conseguenza del rischio sanitario connesso all'infezione da Coronavirus.

3.3.1 Dipartimento della Protezione Civile

Il Dipartimento della Protezione Civile italiano, DPC, ha subito messo a disposizione dei cittadini i dati raccolti ed ha continuato a caricarli giornalmente sul repository github **"Dati COVID-19 Italia"**: `https://github.com/pcm-dpc/COVID-19`.

Tale database è suddiviso in diverse cartelle e si può procedere ad un'esplorazione ed analisi dei dati a livello nazionale ('dati-andamento-nazionale'), regionale ('dati-regioni') o provinciale ('dati-province'). All'interno di queste cartelle troviamo dei file .csv giornalieri e/o cumulativi. Ci focalizziamo sul file 'dpc-covid19-ita-andamento-nazionale.csv', ultimo file della cartella 'dati-andamento-nazionale'. Riportiamo la struttura dei dati in tabella 3.3 e in figura 3.4 ne mostriamo sei campi.

Osservazione 9. Molte colonne includono dati cumulativi, totali. Nell'applicare le PINNs al modello SEIJHDR si è, invece, fatto uso di dati giornalieri; mostriamo come sono stati estratti nella sezione successiva.

	data	totale_positivi	totale_ospedalizzati	deceduti	dimessi_guariti	tamponi
0	2020-02-24T18:00:00	221	127	7	1	4324
1	2020-02-25T18:00:00	311	150	10	1	8623
2	2020-02-26T18:00:00	385	164	12	3	9587
3	2020-02-27T18:00:00	588	304	17	45	12014
4	2020-02-28T18:00:00	821	409	21	46	15695
...	...	...	...	...	...	...
733	2022-02-26T17:00:00	1140558	11866	154416	11437706	186820082
734	2022-02-27T17:00:00	1122278	11601	154560	11487720	187137866
735	2022-02-28T17:00:00	1099934	11565	154767	11528135	187336379
736	2022-03-01T17:00:00	1073230	11164	155000	11601742	187867237
737	2022-03-02T17:00:00	1061610	10635	155214	11651094	188282525

Figura 3.4: Visualizzazione di sei campi del dataset 'dpc-covid19-ita-andamento-nazionale.csv'

3.3.2 Istituto Superiore di Sanità

Il 31/03/21 è stato approvato un accordo di collaborazione scientifica con il quale l'Istituto Superiore di Sanità (ISS) fornisce all'Istituto Nazionale di Fisica Nucleare (INFN) dati anonimi riferiti a ciascun paziente.
Dal sito `https://covid19.infn.it/iss/` sono stati scaricati tre file .csv riguardanti il numero di nuovi casi giornalieri di:

- positivi - 'iss_bydate_italia_positivi.csv';
- ospedalizzati - 'iss_bydate_italia_ricoveri.csv';
- deceduti - 'iss_bydate_italia_deceduti.csv'.

Ciascun dataset è composto da tre colonne descritte in tabella 3.4. Dai tre dataset ne è stato ottenuto uno contenente 4 colonne ('date_of_interest', 'CASE_COUNT', 'DEATH_COUNT' e 'HOSPITALIZED_COUNT'), in analogia con il dataset newyorkese descritto in 3.2 ed infine salvato nel file 'data-by-day-italy.csv', con il seguente codice:

```
import pandas as pd
from datetime import datetime
nuovi_deceduti=pd.read_csv('iss_bydate_italia_deceduti.csv')
nuovi_casi=pd.read_csv('iss_bydate_italia_positivi.csv')
nuovi_ricoveri=pd.read_csv('iss_bydate_italia_ricoveri.csv')

nuovi_deceduti.rename(columns={'casi':'DEATH_COUNT','data':'
    date_of_interest'},inplace=True)
nuovi_casi.rename(columns={'casi':'CASE_COUNT','data':'
    date_of_interest'},inplace=True)
nuovi_ricoveri.rename(columns={'casi':'HOSPITALIZED_COUNT','
    data':'date_of_interest'},inplace=True)

nuovi_deceduti.drop(columns='casi_media7gg', inplace=True)
nuovi_ricoveri.drop(columns='casi_media7gg', inplace=True)
nuovi_casi.drop(columns='casi_media7gg', inplace=True)

nuovi_c_r_d=pd.merge(left=nuovi_casi, right=nuovi_ricoveri,
    on='date_of_interest')
nuovi_c_r_d=pd.merge(left=nuovi_c_r_d,right=nuovi_deceduti,on
    ='date_of_interest')
nuovi_c_r_d['date_of_interest']=pd.to_datetime(nuovi_c_r_d['
    date_of_interest']).dt.strftime('%m/%d/%Y')
nuovi_c_r_d.to_csv('data-by-day-italy.csv')
```

Esempio 3.3. Lettura dataset 'data-by-day-italy', salvato con il precedente codice come un DataFrame Pandas, in un range di date che va dal 7 al 14 gennaio del 2022. Mostriamo tale risultato nella figura 3.5.

	date_of_interest	CASE_COUNT	HOSPITALIZED_COUNT	DEATH_COUNT
687	01/07/2022	205382	1600	186
688	01/08/2022	136813	1558	218
689	01/09/2022	63127	1287	215
690	01/10/2022	220913	1593	227
691	01/11/2022	180024	1553	270
692	01/12/2022	162733	1570	243
693	01/13/2022	152685	1494	262
694	01/14/2022	156972	1474	278

Figura 3.5: Visualizzazione dei dati presenti in 'data-by-day-italy' relativi alla seconda settimana di gennaio 2022

3.3.3 Vaccini anti-COVID-19

Nel corso del 2021 il commissario straordinario per l'emergenza Covid-19 mette a disposizione il repository github **Covid-19 Opendata Vaccini**: `https://github.com/italia/covid19-opendata-vaccini`, gestito dalla comunità del governo italiano 'Developers Italia'.
Esso contiene dati accessibili a chiunque su consegna e somministrazione dei vaccini anti COVID-19 in Italia. Analizziamo il file 'somministrazioni-vaccini-summary-latest.csv', contenuto nella cartella 'Dati'. E' composto da una tabella con 16 campi, descritti in tabella 3.5. Nel grafico 3.6 mostriamo l'andamento delle somministrazioni a livello nazionale. Esso riguarda il campo `'new_vacc_7davg'` aggiunto al file 'somministrazioni-vaccini-summary-latest.csv', ottenuto sommando, giornalmente, i campi `'prima_dose'`, `'seconda_dose'`, `'dose_addizionale_booster'` e `'pregressa_infezione'` ed applicando infine una media mobile a 7 giorni.

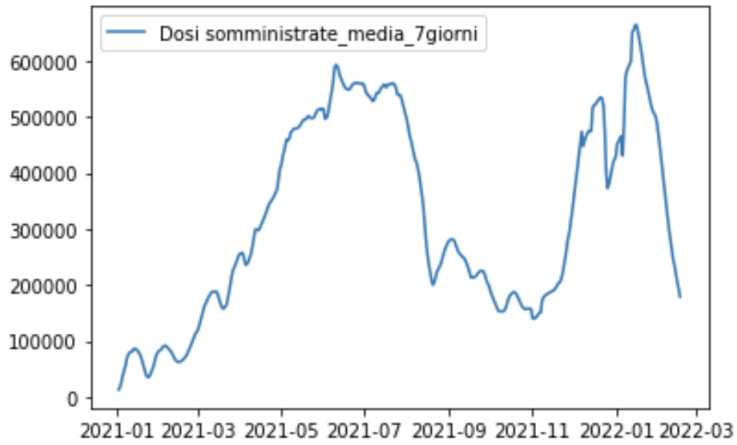

Figura 3.6: Andamento media a 7 giorni delle dosi di vaccino anti-COVID19 somministrate in Italia dal 27/12/20 al 17/02/22

3.4 Le varianti

Per avere una visione completa sui dati inerenti la malattia COVID-19 è opportuno parlare delle mutazioni del SARS-CoV-2, il virus che la causa. Un virus con una o più nuove mutazioni viene indicato come una "variante" del virus originale [33].

3.4.1 Classificazione

Dalla fine del 2019, attraverso il sequenziamento di genomi di SARS-CoV-2, sono state individuate migliaia di varianti, raggruppate in gruppi più grandi come lignaggi o clade. Le varianti del SARS-CoV-2 vengono distinte dal WHO, *World Health Organization*, noto in italiano come OMS (Organizzazione Mondiale di Sanità) in tre categorie [34]:

- **VUM** - Variante sotto monitoraggio - una variante SARS-CoV-2 con cambiamenti genetici che si sospetta possano influenzare le caratteristiche del virus con qualche indicazione che possa rappresentare un rischio futuro, ma l'evidenza dell'impatto fenotipico o epidemiologico non è attualmente chiara, e richiede un monitoraggio rafforzato e una valutazione ripetuta in attesa di nuove prove;

- **VOI** - Variante di interesse - una variante con con cambiamenti genetici che sono previsti o noti per influenzare le caratteristiche del virus come la trasmissibilità, la gravità della malattia, la fuga immunitaria,

la fuga diagnostica o terapeutica.
É identificata per causare una significativa trasmissione comunitaria o cluster multipli di COVID-19, in più paesi con una prevalenza relativa crescente insieme a un numero crescente di casi nel tempo, o altri impatti epidemiologici evidenti che suggeriscano un rischio emergente per la salute pubblica globale.

- **VOC** - Variante preoccupante - ovvero una variante che soddisfa la definizione di VOI e, attraverso una valutazione comparativa, è stato dimostrato che è associata a uno o più dei seguenti cambiamenti a un livello di rilevanza globale per la salute pubblica:
  - aumento della trasmissibilità o cambiamento dannoso nell'epidemiologia del COVID-19;
  - aumento della virulenza o cambiamento nella presentazione clinica della malattia;
  - diminuzione dell'efficacia delle misure sociali e di salute pubblica o delle diagnosi, dei vaccini e delle terapie disponibili.

L'OMS può poi segnalare ai Paesi eventuali interventi da mettere in atto per prevenire la diffusione di quella variante.

3.4.2 Nomenclatura

I sistemi di nomenclatura per nominare e tracciare i lignaggi di SARS-CoV-2 sono stati stabiliti da **GISAID** [35], **Nextstrain** [36] e **Pango** [37]. L'OMS ha convocato un gruppo di scienziati per considerare etichette facili da pronunciare e non stigmatizzanti per VOI e VOC. Tale gruppo ha raccomandato di usare lettere dell'alfabeto greco, che saranno più facili e pratiche da discutere per un pubblico non scientifico.
Al 18/02/2022 sono cinque le VOCs definite dall'OMS:

1. Variante **Alfa** (nota anche come B.1.1.7 nel sistema Pango) identificata per la prima volta nel Regno Unito a settembre 2020;

2. Variante **Beta** (nota anche come B.1.351 nel sistema Pango) identificata in Sud Africa a maggio 2020;

3. Variante **Gamma** (denominata P.1 nel sistema Pango) identificata in Brasile a novembre 2020;

4. Variante **Delta** (nota anche come B.1.617 nel sistema Pango) rilevata per la prima volta in India ad ottobre del 2020;

5. Variante **Omicron** (denominata B.1.1.529 nel sistema Pango) rilevata in molte nazioni a novembre 2021. Attualmente predominante in Italia ed Europa.

I dati sulle varianti sono stati scaricati dalla banca dati EpiCoV di GISAID [38], *Global Initiative on Sharing Avian Influenza Data*, in due date: 01/10/21 e 20/02/22, sotto forma di file .xlsx. Essi contengono diversi fogli, uno per ogni VOC e uno per ogni VOI. Ogni foglio contiene una tabella con i conteggi dei genomi di hCoV-19 inviati al GISAID, relativi a quella VOC/VOI, aggregati per Paese e per settimana ("**count**"). Un altro campo ("**total**") relativo al numero totale di *submissions* (invii), viene visualizzato solo se c'è un invio con la variante data, altrimenti viene lasciato vuoto. Nelle due date di riferimento le varianti predominanti erano la Delta e l'Omicron.

3.4.3 Delta

Analizziamo la diffusione della variante Delta in Italia e UK dalla 31ª settimana del 2020 (20w31) alla 39ª settimana del 2021 (21w39), rispetto al numero di genomi hCoV-19 inviati alla GISAID.

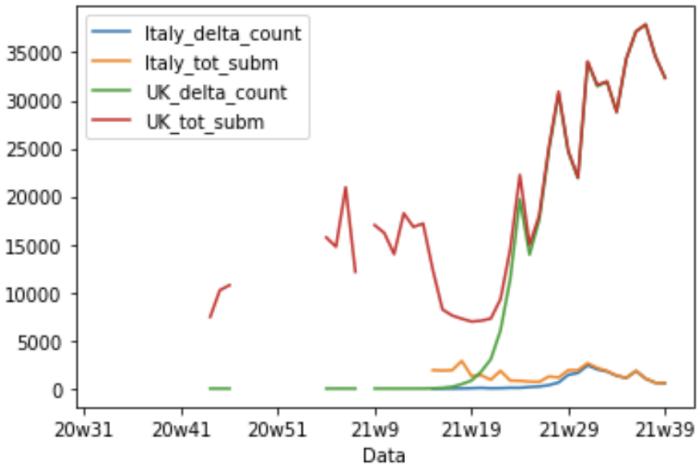

Figura 3.7: Grafico conteggi settimanali variante delta Italia e UK (azzurro e verde) e conteggi settimanali di tutte le varianti dei genomi hCoV-19 inviati alla GISAID da parte di Italia e UK (arancione e rosso)

Il grafico in figura 3.7 è stato ottenuto eseguendo il seguente script Python:

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
```

```
file_name="csv/
    gisaid_variants_by_week_2021_10_01_0831_2021_10_01.xlsx"
df=pd.read_excel(file_name)
xl_file = pd.ExcelFile(file_name)
df = {sheet_name: xl_file.parse(sheet_name)
          for sheet_name in xl_file.sheet_names}
df_delta=df['VOC Delta']
delta_it_count=df_delta.loc[150,:] #150 e 151 sono gli indici
     di riga corrispondenti all'Italia
delta_it_tot=df_delta.loc[151,:]
delta_uk_count=df_delta.loc[320,:]#320 e 321 sono gli indici
    di riga corrispondenti all'UK
delta_uk_tot=df_delta.loc[321,:]
df_uk_it_delta=pd.DataFrame({'Italy_delta_count':
    delta_it_count,'Italy_tot_subm':delta_it_tot,'
    UK_delta_count':delta_uk_count,'UK_tot_subm':delta_uk_tot
    })
df_uk_it_delta=df_uk_it_delta.iloc[2:,:] #eliminazione delle
    prime due righe
l=['19w51','19w52']
suffix_str = "20w"
l=l+ [suffix_str+str(n) for n in range(1,53)]+['21w'+str(n)
    for n in range(1,42)]
df_uk_it_delta['Data']=l
df_uk_it_delta=df_uk_it_delta.set_index('Data')
df_uk_it_delta=df_uk_it_delta.iloc[32:93,:]#eliminazione
    prime 32 e ultime 2 righe in quanto i valori iniziali
    erano tutti Nan e i dati delle ultime settimane non sono
    completi
df_uk_it_delta.plot()
```

Osservazione 10. La curva in rosso in figura 3.7 presenta una decrescita solo perchè sono stati inviati meno sequenziamenti di variante Delta alla GISAID, rispetto alle settimane precedenti. Studiamo dunque la percentuale con la quale è presente una certa variante rispetto a tutti i sequenziamenti inviati alla GISAID.

Aggiungiamo due campi al dataframe `'df_uk_it_delta'` definito nello script precedente e mostriamo le curve risultanti in due grafici (figura 3.8) con il seguente script Python:

```
df_uk_it_delta['Italy_percentage_delta_over_submissions']=
    df_uk_it_delta['Italy_delta_count']/df_uk_it_delta['
    Italy_tot_subm']*100
df_uk_it_delta['uk_percentage_delta_over_submissions']=
    df_uk_it_delta['UK_delta_count']/df_uk_it_delta['
    UK_tot_subm']*100
ax1 = plt.subplot(1, 2, 1)
```

```
plt.plot(df_uk_it_delta.index,df_uk_it_delta.
    Italy_percentage_delta_over_submissions)
plt.xlabel("settimane")
plt.ylabel("% Variante delta in Italia")
plt.title("Andamento in % della variante Delta\n in Italia su
     tutti i sequenziamenti")
x=range(0,61,5)
index=df_uk_it_delta.index
y=index[0:61:5]
plt.xticks(x, y, rotation=60)

plt.subplot(1, 2, 2, sharey=ax1)
plt.plot(df_uk_it_delta.index,df_uk_it_delta.
    uk_percentage_delta_over_submissions)
plt.xlabel("settimane")
plt.ylabel("% Variante delta in UK")
plt.title("Andamento in % della variante Delta\n in UK su
    tutti i sequenziamenti")
x=range(0,61,5)
index=df_uk_it_delta.index
y=index[0:61:5]
plt.xticks(x, y, rotation=60)
plt.subplots_adjust(right=1.5)

plt.show()
```

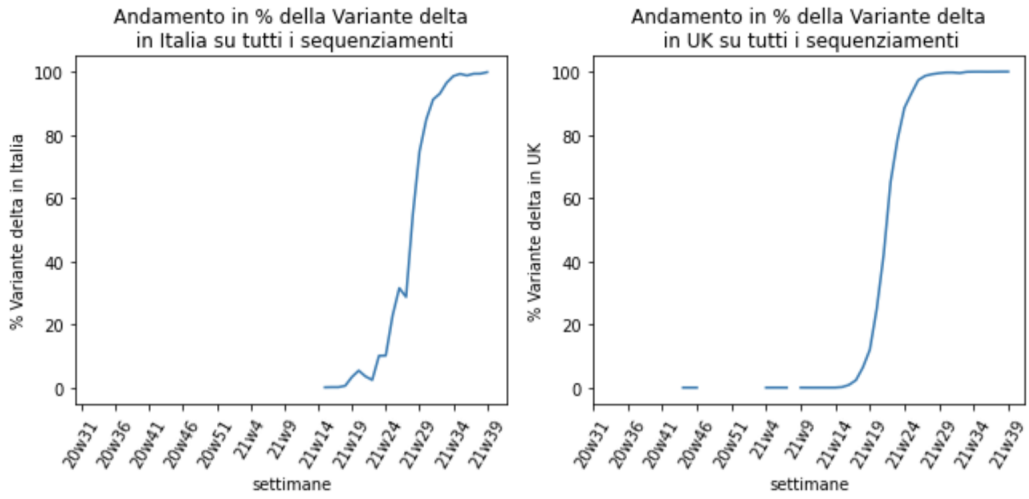

Figura 3.8: Andamento in percentuale della variante Delta in Italia (a sinistra) e in UK (a destra) nel periodo compreso tra la 31ª settimana del 2020 e la 39ª settimana del 2021

Osservazione 11. Notiamo che la variante Delta:

- in Italia é stata segnalata la prima volta alla GISAID intorno alla 15ª settimana del 2021 ed è risultata prevalente su tutti i sequenziamenti della 34ª settimana del 2021.
- in UK è prevalente intorno alla 29ª settimana del 2021.

3.4.4 Omicron

Esaminiamo infine la percentuale di casi Covid19 di variante Omicron inviati alla GISAID dall'Italia dalla prima settimana del 2021 alla sesta settimana del 2022.

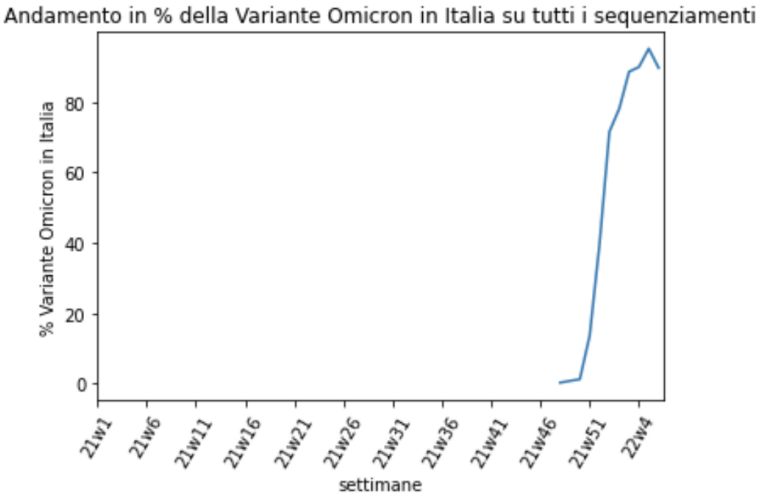

Figura 3.9: Andamento in percentuale della variante Omicron in Italia nel periodo compreso tra la prima settimana del 2021 e la sesta settimana del 2022

Osservazione 12. Dalla figura 3.9 è evidente la predominanza di tale variante rispetto alle altre nelle prime settimane del 2022.

NOME DELLA METRICA	DESCRIZIONE
data	Data di riferimento dei dati, formato YYYY-MM-DD HH:MM:SS (ISO 8601) Ora italiana
stato	Stato di riferimento
ricoverati_con_sintomi	Ricoverati con sintomi
terapia_intensiva	Ricoverati in terapia intensiva
totale_ospedalizzati	Totale ospedalizzati
isolamento_domiciliare	Persone in isolamento domiciliare
totale_positivi	totale_ospedalizzati+ isolamento_domiciliare
variazione_totale_positivi	totale_positivi giorno corrente - totale_positivi giorno precedente
nuovi_positivi	totale_casi giorno corrente - totale_casi giorno precedente
dimessi_guariti	Persone dimesse guarite
deceduti	Persone decedute
casi_da_sospetto_diagnostico	Casi positivi al tampone emersi da attività clinica. (Non più popolato)
casi_da_screening	Casi positivi emersi da indagini e test, pianificati a livello nazionale o regionale. (Non più popolato)
totale_casi	Totale casi positivi
tamponi	tamponi_test_molecolare + tamponi_test_antigenico_rapido
casi_testati	Totale dei soggetti sottoposti al test
note	Note
ingressi_terapia_intensiva	Ingressi giornalieri in terapia intensiva
note_test	Note sui test effettuati
note_casi	Note sui casi testati
totale_positivi_test_molecolare	Totale dei soggetti positivi al test molecolare
totale_positivi_test_ antigenico_rapido	Totale dei soggetti positivi al test antigenico rapido
tamponi_test_molecolare	Totale tamponi processati con test molecolari
tamponi_test_antigenico_rapido	Totale tamponi processati con test antigenico rapito

Tabella 3.3: Descrizione dei campi del database 'dpc-covid19-ita-andamento-nazionale.csv'

NOME DELLA METRICA	DESCRIZIONE
data	data di riferimento, formato aaaa-mm-gg
casi	numero di casi giornalieri di positivi/ ricoveri/deceduti riferiti rispettivamente alla data del tampone/ricovero/decesso
casi_media7gg	andamento della media calcolata su 7 giorni (media mobile) dei 'casi', riferita al giorno centrale dei 7 giorni considerati

Tabella 3.4: Descrizione dei campi dei database in 3.3.2

NOME DEL CAMPO	DESCRIZIONE
index	Codice identificativo del record
area	Sigla della regione in cui è avvenuta la somministrazione
data_somministrazione	Giorno in cui è avvenuta la somministrazione, formato aaaa-mm-gg
totale	Numero totale di dosi di vaccino somministrate per giorno e regione
sesso_maschile	Totale dei soggetti di sesso maschile a cui è stato somministrato il vaccino per giorno e regione
sesso_femminile	Totale dei soggetti di sesso femminile a cui è stato somministrato il vaccino per giorno e regione
prima_dose	Numero prime somministrazioni
seconda_dose	Numero seconde somministrazioni
pregressa_infezione	Numero di somministrazioni effettuate a soggetti con pregressa infezione da covid-19 nel periodo 3-6 mesi e che, pertanto, concludono il ciclo vaccinale con un'unica dose
dose_addizionale_booster	Numero somministrazioni dose addizionale/richiamo
codice_NUTS1	Classificazione europea delle unità territoriali NUTS: livello NUTS 1
codice_NUTS2	Classificazione europea delle unità territoriali NUTS: livello NUTS 2
codice_regione_ISTAT	Codice ISTAT della Regione
nome_regione	Denominazione standard dell'area

Tabella 3.5: Descrizione dei campi del database 'somministrazioni-vaccini-summary-latest.csv'

Capitolo 4

Risultati numerici

In questo capitolo vengono riportati i risultati numerici ottenuti applicando le PINNs al modello SEIJDHR. Viene dapprima data un'idea dei codici utilizzati e infine vengono mostrati i grafici di rilievo ottenuti dando in input:

- il file 'data-by-day.csv', contenente dati newyorkesi, analizzato in 3.2;
- il file 'data-by-day-italy.csv', contenente dati italiani, costruito in 3.3.2.

Per il secondo ci soffermeremo sulle modifiche che è stato opportuno effettuare ai codici.

4.1 Codici

I risultati numerici sono stati ottenuti eseguendo, con le opportune modifiche, tre codici python reperiti da [5] seguendo il *path* 'PINN-COVID-master/ ModelUncertainty-NYCData/Iteger_Order_Models_I1-I2-I3-T1':

1. 'Model1_training_Vacc.py';
2. 'Model1_PostProcess.py';
3. 'Model1_Prediction.py'.

dove con 'Model1' e 'I1' viene indicato il modello SEIJDHR.

In 1 viene definita la `class PhysicsInformedNN`. Il metodo `__init__` di tale classe, oltre all'istanza self, ha come attributi:
`'t_train', 'I_new_train', 'D_new_train', 'H_new_train', 'I_sum_train', 'D_sum_train', 'H_sum_train', 'U0', 't_f', 'lb', 'ub', 'N', 'layers',`

'layers_beta', 'layers_p', 'layers_q', 'sf'.

All'interno della classe sono definite molte funzioni, tra cui `initialize_NN` per inizializzare la rete neurale e `neural_net` per definirne l'architettura, più volte richiamata nel codice stesso:

```
    def neural_net(self, t, weights, biases):
        num_layers = len(weights) + 1

        H = 2.0*(t-self.lb)/(self.ub-self.lb) - 1.0
        for l in range(0,num_layers-2):
            W = weights[l]
            b = biases[l]
            H = tf.tanh(tf.add(tf.matmul(H, W), b))
        W = weights[-1]
        b = biases[-1]
        Y = tf.add(tf.matmul(H, W), b)
        return Y
```

Per ottenere i compartimenti del modello viene definita la funzione 'net_u':

```
    def net_u(self, t):
        SEIJDHR = self.neural_net(t, self.weights, self.
    biases)
        E = SEIJDHR[:,0:1]
        I = SEIJDHR[:,1:2]
        J = SEIJDHR[:,2:3]
        D = SEIJDHR[:,3:4]
        H = SEIJDHR[:,4:5]
        R = SEIJDHR[:,5:6]
        I_sum = SEIJDHR[:,6:7]
        H_sum = SEIJDHR[:,7:8]
        D_sum = D
        S = self.N-E-I-J-D-H-R
        return S, E, I, J, D, H, R, I_sum, D_sum, H_sum
```

Il modello viene invece definito nella funzione `net_f`:

```
def net_f(self, t):
        .
        .
        .
        f_E = E_t - (betaI * (I + eps1 * J + eps2 * H) / self
    .N) * S + alpha * E
        f_I = I_t - (delta * alpha) * E + gamma * I
        f_J = J_t - ((1 - delta) * alpha) * E + gammaA * J
        f_D = D_t - (q * phiD) * H
        f_H = H_t - (p * gamma) * I + (q * phiD) * H + ((1 -
    q) * phiR) * H
        f_R = R_t - gammaA * J - ((1 - p) * gamma) * I - ((1
    - q) * phiR) * H - VCn * S / self.N
```

```
11          f_I_sum = I_sum_t - (delta * alpha) * E
12          f_H_sum = H_sum_t - (p * gamma) * I
13      return f_E, f_I, f_J, f_D, f_H, f_R, f_I_sum, f_H_sum
```

I valori dei parametri fissi del modello SEIJDHR (1.19), il cui significato è stato descritto nella tabella 1.1, sono riportati in tabella 4.1.

PARAMETRO	VALORE
ϵ	0.75
δ	0.6
$1/\alpha$	5.2
$1/\gamma_a$	6
$1/\gamma$	6
$1/\phi_R$	7.5
$1/\phi_D$	15

Tabella 4.1: Valore dei parametri fissi del modello (1.19)

I parametri dipendenti dal tempo $\beta_I(t)$, $p(t)$ e $q(t)$ vengono dedotti attraverso tre reti neurali allenate in 1 e fissati al loro ultimo valore per la fase predittiva 3.
Vengono inoltre considerati due diversi scenari di vaccinazione considerando due tassi di vaccinazione ($Vac.rate$) nel calcolare la **vaccinazione effettiva** al giorno t ($V(t)$), dove:

$$V(t) = \begin{cases} 0, & t \in [0, t_{Vac.start}), \\ (Vac.rate) \cdot (t - t_{Vac.start}), & t \in [t_{Vac.start}, t_{Vac.cap}), \\ Vac.cap, & t \geq t_{Vac.cap}. \end{cases} \tag{4.1}$$

$Vac.cap$ e $t_{Vac.start}$ rappresentano rispettivamente il valore di saturazione della vaccinazione effettiva e il giorno di inizio della vaccinazione effettiva.

In 1 vengono definite quattro funzioni costo, `lossU0, lossU, lossF, loss`:

```
self.lossU0 = tf.reduce_mean(tf.square(self.E0_u - self.
    E0_pred)) + \
                    tf.reduce_mean(tf.square(self.I0_u -
    self.I0_pred)) + \
                    tf.reduce_mean(tf.square(self.J0_u -
    self.J0_pred)) + \
                    tf.reduce_mean(tf.square(self.D0_u -
    self.D0_pred)) + \
```

```
                        tf.reduce_mean(tf.square(self.H0_u -
    self.H0_pred)) + \
                        tf.reduce_mean(tf.square(self.R0_u -
    self.R0_pred))

self.lossU = 120 * tf.reduce_mean(tf.square(self.I_new_u[:-1,
     :] - self.I_new_pred)) + \
                    10 * 120 * tf.reduce_mean(tf.square(self
    .D_new_u[:-1, :] - self.D_new_pred)) + \
                    3 * 120 * tf.reduce_mean(tf.square(self.
    H_new_u[:-1, :] - self.H_new_pred)) + \
                    tf.reduce_mean(tf.square(self.I_sum_u -
    self.I_sum_pred)) + \
                    10 * tf.reduce_mean(tf.square(self.
    D_sum_u - self.D_sum_pred)) + \
                    3 * tf.reduce_mean(tf.square(self.
    H_sum_u - self.H_sum_pred))

self.lossF = tf.reduce_mean(tf.square(self.E_f)) + tf.
    reduce_mean(tf.square(self.I_f)) + \
                    tf.reduce_mean(tf.square(self.J_f)) + tf
    .reduce_mean(tf.square(self.D_f)) +\
                    tf.reduce_mean(tf.square(self.H_f)) + tf
    .reduce_mean(tf.square(self.R_f)) + \
                    tf.reduce_mean(tf.square(self.I_sum_f))
    + tf.reduce_mean(tf.square(self.H_sum_f))

self.loss = self.lossU0 + self.lossU + self.lossF
```

L'ultima funzione di perdita `self.loss` è poi minimizzata con 2 algoritmi:

- l'algoritmo `'Adam'` (*ADAptive Moment estimation*), un'estensione dell'algoritmo di discesa stocastica del gradiente; per approfondire rimandiamo a [39];
- l'algoritmo `'L-BFGS-B'`, un metodo di ottimizzazione quasi-Newton, estensione del più noto BFGS (Broyden, Fletcher, Goldfarb e Shanno); per approfondire rimandiamo a [40].

Vengono così definiti:

```
self.optimizer = tf.contrib.opt.ScipyOptimizerInterface(self.
    loss, method='L-BFGS-B',
         options={'maxiter': 50000,
                  'maxfun': 50000,
                  'maxcor': 50,
                   'maxls': 50,
                   'ftol': 1.0 * np.finfo(float).eps})

```

```
self.optimizer_Adam = tf.train.AdamOptimizer()
self.train_op_Adam = self.optimizer_Adam.minimize(self.loss)
```

In 2 viene eseguita una media dei dati ottenuti in seguito a undici allenamenti del modello e i risultati vengono salvati nella cartella `'Train-Results-mm-gg-Average'`.

4.2 Previsioni NYC

Assumiamo fissa la popolazione della città di New York, pari a $N = 8399000$.
Ai dati in input, 'data-by-day.csv', con range di date 29/02/2020 - 21/03/2021, viene applicata una media mobile a 7 giorni e rimossi i primi sei giorni.
La rete viene dunque allenata su dati che riguardano il periodo di tempo `'t_train'`= 06/03/2020 - 21/03/2021.
Vengono poi prodotte previsioni nel periodo `'t_pred'`= 20/03/2021 - 30/12/2021.

Le vaccinazioni effettive giornaliere vengono approssimate con la funzione:

$$V(t) = \begin{cases} 0, \, t < 300, \\ 600(t - 300), \, t < 350 \\ 30000, \, t \geq 350 \end{cases} \tag{4.2}$$

dove il primo valore è assunto 0 prima del primo gennaio 2021, in quanto la prima somministrazione di vaccino a New York è avvenuta il 14/12/2020 e si considerano due settimane di ritardo affinchè si acquisisca l'immunità.

Per i dati newyorkesi la rete è stata inizializzata con pesi di xavier, definiti nella funzione `xavier_init`. Essa è poi stata richiamata in `inizialize_NN`:

```
def initialize_NN(self, layers):
        weights = []
        biases = []
        num_layers = len(layers)
        for l in range(0, num_layers - 1):
            W = self.xavier_init(size=[layers[l], layers[l + 1]])  # weights for the current layer
            b = tf.Variable(tf.zeros([1, layers[l + 1]], dtype=tf.float64),
                            #dtype=tf.float64)  # biases for the current layer
```

```
            weights.append(W)  # save the elements in W to
    weights (a row vector)
            biases.append(b)  # save the elements in b to
    biases (a 1Xsum(layers) row vector)
        return weights, biases
```

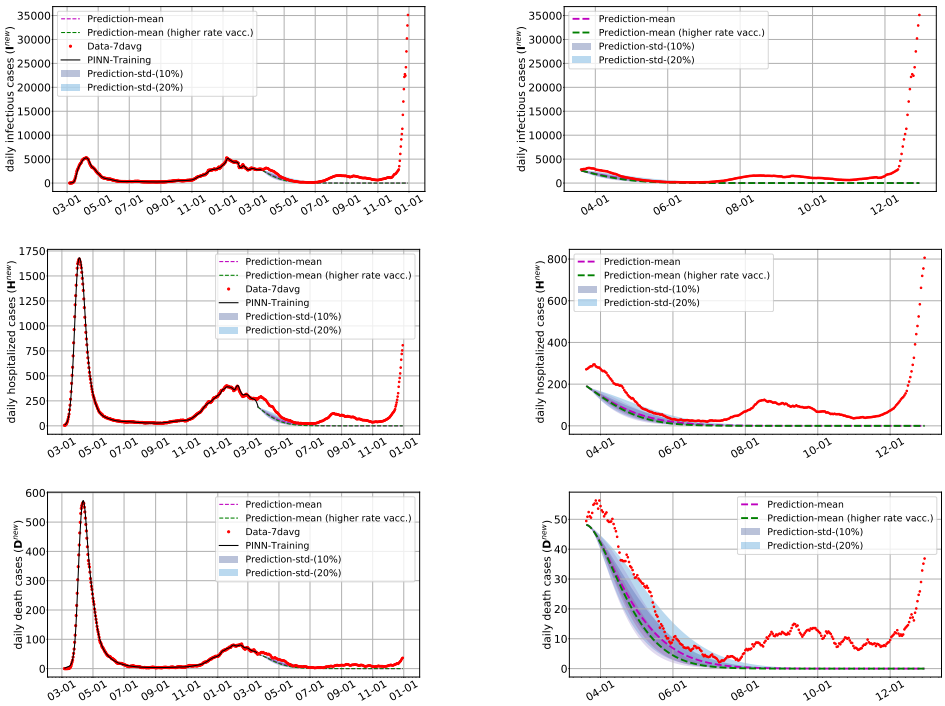

Figura 4.1: Nelle ascisse viene riportato il mese e il giorno (mm-gg). Nelle ordinate, invece, dall'alto verso il basso: il numero di nuovi infetti, ospedalizzati e deceduti giornalieri. A destra viene mostrato uno zoom sulla parte predittiva (dal 20/03/21 al 30/12/21)

Osservazione 13. Dalla figura 4.1 notiamo un'errata previsione del numero di nuovi casi e deceduti giornalieri tra novembre 2021 e gennaio 2022. Notiamo che tal periodo coincide con la diffusione esponenziale della variante Omicron.

Le tre reti per i parametri $\beta_I(t)$, $p(t)$ e $q(t)$ sono così definite:

```
    def net_BetaI(self,t):
        BetaI = self.neural_net(t, self.weights_beta, self.
    biases_beta)
        bound_b = [tf.constant(0.05, dtype=tf.float64), tf.
    constant(1.0, dtype=tf.float64)]
        return bound_b[0]+(bound_b[1]-bound_b[0])*tf.sigmoid(
    BetaI)
```

```
5
6       def net_p(self, t):
7           p = self.neural_net(t, self.weights_p, self.biases_p)
8           return tf.sigmoid(p)
9
10      def net_q(self,t):
11          q = self.neural_net(t, self.weights_q, self.biases_q)
12          return 0.15+(0.6-0.15)*tf.sigmoid(q)
```

4.3 Previsioni Italia

Fissiamo la popolazione a $N = 60461826$.
Mostriamo i risultati ottenuti per diverse funzioni $V(t)$, in diversi periodi di tempo `'t_train'` e `'t_pred'`.

Prima scelta

Come prima scelta si è deciso di non modificare le condizioni iniziali del modello, i periodi di allenamento e di previsione della rete:
`'t_train'`=06/03/2020-21/03/2021.
`'t_pred'`=20/03/2021 - 30/12/2021.

É stato invece necessario modificare la funzione sulle vaccinazioni effettive:

$$V(t) = \begin{cases} 0,\ t < 313, \\ 1754(t - 313), t < 370 \\ 100000, t \geq 370 \end{cases} \tag{4.3}$$

dove $t_{Vac.start} = 313$ in quanto in Italia la prima dose di vaccino è stata somministrata esattamente tredici giorni dopo rispetto a NYC.

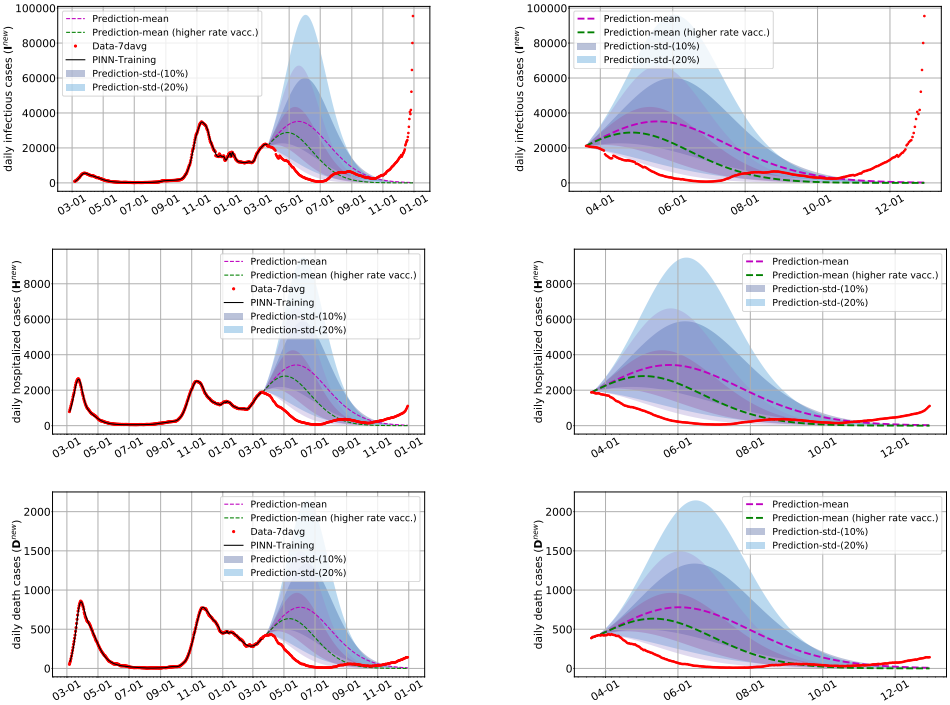

Figura 4.2: Nelle ascisse viene riportato il mese e il giorno (mm-gg). Nelle ordinate, invece, dall'alto verso il basso: il numero di nuovi infetti, ospedalizzati e deceduti giornalieri. A destra viene mostrato uno zoom sulla parte predittiva (dal 20/03/21 al 30/12/21)

Osservazione 14. Dalla figura 4.9 notiamo delle incongruenze soprattutto nei primi due grafici in alto. Il numero di nuovi casi infetti giornalieri ha avuto una crescita esponenziale, nonostante le vaccinazioni, a partire da novembre 2021. Ciò è stato dovuto all'arrivo della variante Omicron in Italia. Si nota, inoltre, una sottostima del modello rispetto ai dati reali.

Seconda scelta

Alleniamo adesso la PINN su dati riferiti ad un periodo di tempo di tre mesi: `'t_train'`= 1/10/21 - 31/12/21.
`'t_pred'`=30/12/21-31/01/22
Inoltre sono state modificate due delle sei condizioni iniziali:

- H_0 = primo valore di `HOSPITALIZED_COUNT` del nuovo database di input
- R_0 = numero di guariti dei 4 mesi precedenti all'1/10/21 in quanto in tale data il rischio di reinfezione per tale individui è basso.

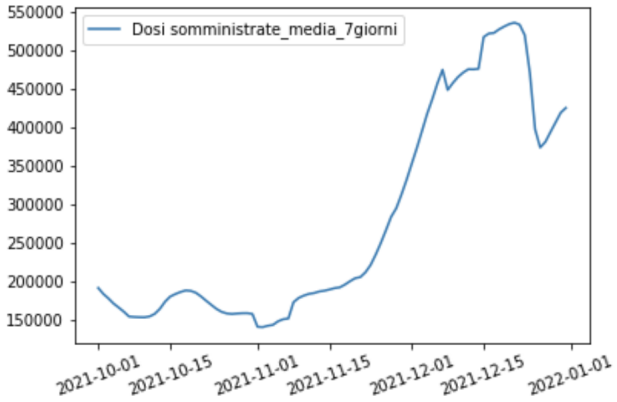

Figura 4.3: Media a 7 giorni delle somministrazioni di vaccini in Italia nel periodo 1/10/21 - 31/12/21

Sulla base della figura 4.3 si è quindi approssimata la funzione $V(t)$:

$$V(t) = \begin{cases} 170000, t < 30, \\ 350000, 30 < t < 60 \\ 450000, t > 60 \end{cases} \tag{4.4}$$

Nell'eseguire il codice 3 con le nuove condizioni iniziali si è notato un punto di cuspide nel grafico dei nuovi deceduti giornalieri, riportato in figura 4.4.

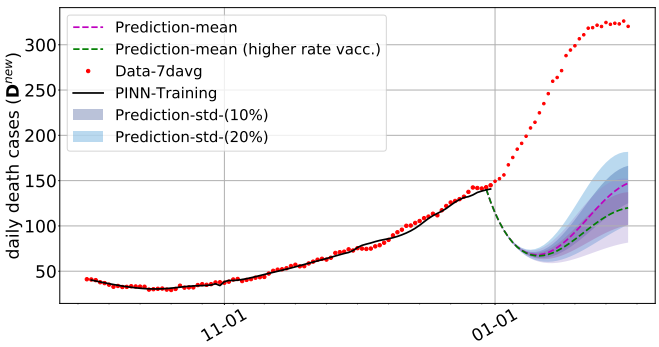

Figura 4.4: Errore nella previsione del numero di nuovi deceduti giornalieri

Si è visto che, modificando opportunamente la funzione `'net_p'`, il punto di cuspide svaniva. Si è dunque effettuata la seguente modifica:

```
    def net_p(self, t):
        p = self.neural_net(t, self.weights_p, self.biases_p)
        return 0.004+tf.sigmoid(p)
```

Riportiamo i risultati ottenuti in figura 4.5.

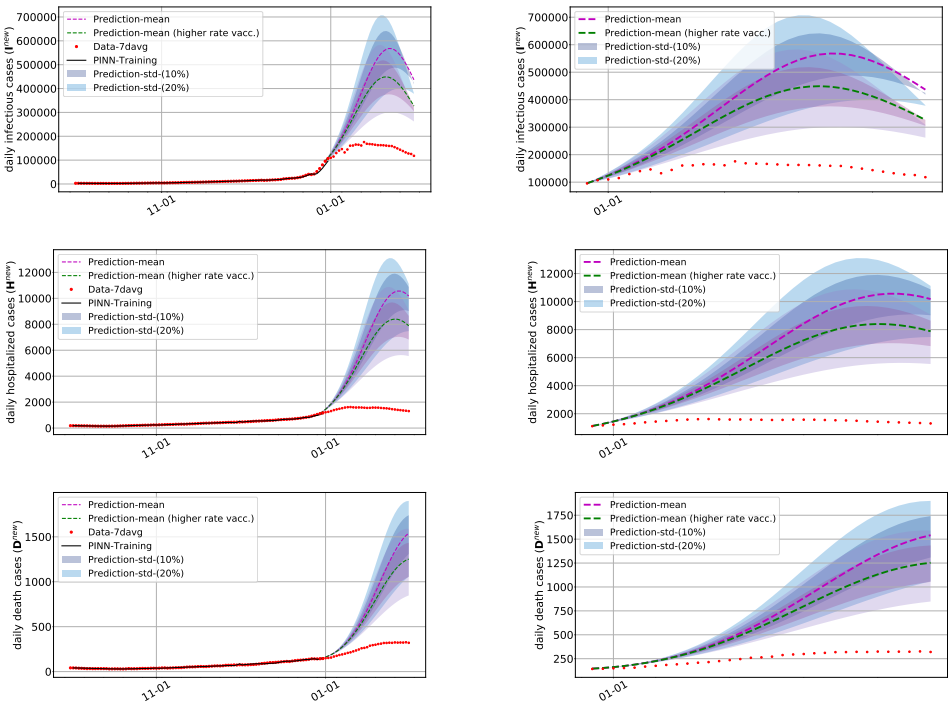

Figura 4.5: Nelle ascisse viene riportato il mese e il giorno (mm-gg). Nelle ordinate, invece, dall'alto verso il basso: il numero di nuovi infetti, ospedalizzati e deceduti giornalieri. A destra viene mostrato uno zoom sulla parte predittiva (dal 30/12/21 al 31/01/22)

Osservazione 15. Dalla figura 4.5 si nota una sovrastima dei tre compartimenti I, H e D nel periodo di previsione.

Terza scelta

Dalla figura 4.3 si è notato un andamento decrescente a fine dicembre fino al raggiungimento di un minimo, probabilmente dovuto ad un rallentamento nelle somministrazioni durante il periodo di festività. É stato dunque modificato l'intervallo di giorni per l'allenamento della rete in `'t_train'`=1/10/21 - 22/12/21 e il periodo di previsione in `'t_pred`=21/12/21 - 12/01/22.
Le condizioni iniziali del modello sono state lasciate invariate rispetto alla seconda scelta.
Le somministrazioni di vaccino in Italia nel periodo di allenamento della rete riferito alla terza scelta viene schematizzato in figura 4.6.

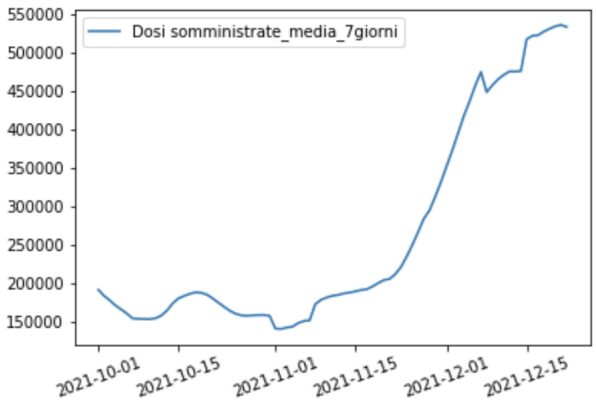

Figura 4.6: Media a 7 giorni delle somministrazioni di vaccini in Italia nel periodo 1/10/21 - 22/12/21

Dalla figura 4.6 si nota un cambio di trend dopo circa 50 giorni, quindi la funzione vaccinazione effettiva è stata approssimata con due tratti:

$$V(t) = \begin{cases} 169033, t < 51, \\ 11580t - 375944, t \geq 51. \end{cases} \tag{4.5}$$

Il primo valore è dato dalla media di tutti i valori di somministrazioni di vaccino riferiti al periodo 1/10/21 - 19/11/21. Il secondo è una funzione ottenuta come regressione lineare dei valori di partenza. Mostriamo come sono stati trovati i valori del coefficiente angolare della retta e dell'intercetta e infine un grafico esplicativo 4.7:

```
X_regr=np.array(list(range(51,84)))
y_regr=np.array(list(vaccination_df['new_vacc_7davg'].loc[
    datarange][50:]))
X_regr =X_regr.reshape(33,1)
regr = linear_model.LinearRegression()
regr.fit(X_regr, y_regr)
print('Coefficiente angolare: ',regr.coef_)
print('Intercetta: ',regr.intercept_)
plt.scatter(X_regr, y_regr,  color='tab:blue')
plt.plot(X_regr, regr.coef_*X_regr+regr.intercept_, color='
    red', linewidth=3)
```

```
Coefficiente angolare:  [11580.13579068]
Intercetta:  -375944.1326075885
```

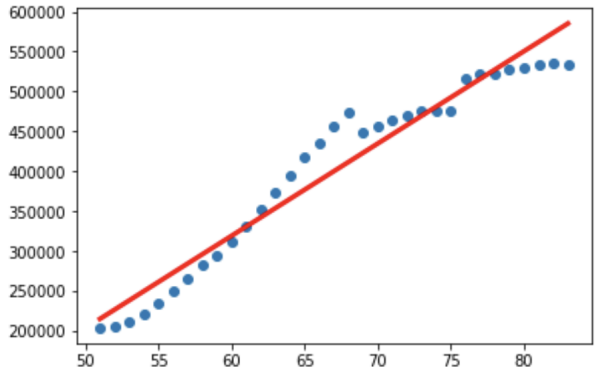

Figura 4.7: In blu i valori giornalieri sul numero di somministrazioni di vaccino; in rosso la retta che approssima tali valori. Il periodo di riferimento va dal 20/11/21 al 22/12/21 ed è stato trasformato in un range numerico da 51 ad 83

Come per la seconda scelta, il grafico dei nuovi deceduti nel periodo di previsioni (figura 4.8) mostra un cambiamento di trend, privo di spiegazione.

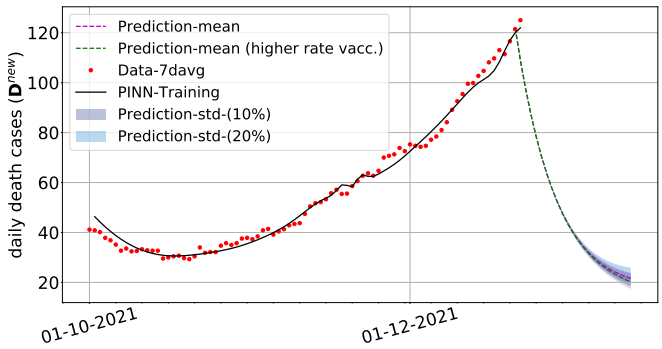

Figura 4.8: Errore nella previsione del numero di nuovi deceduti giornalieri

Si è allora modificata la rete del parametro p, l' inizializzazione dei pesi, usando la tecnica di Xavier Glorot uniforme [41]:

```
        def xavier_init(self, size):
            in_dim = size[0]
            out_dim = size[1]
            limit = np.sqrt(6 / (in_dim + out_dim))
            return tf.Variable(tf.random.uniform([in_dim,
  out_dim], minval=-limit, maxval=limit, dtype=tf.float64),
```

```
                                dtype=tf.float64)

        def net_p(self, t):
            p = self.neural_net(t, self.weights_p, self.
    biases_p)
        return 0.004+tf.nn.relu(p)
```

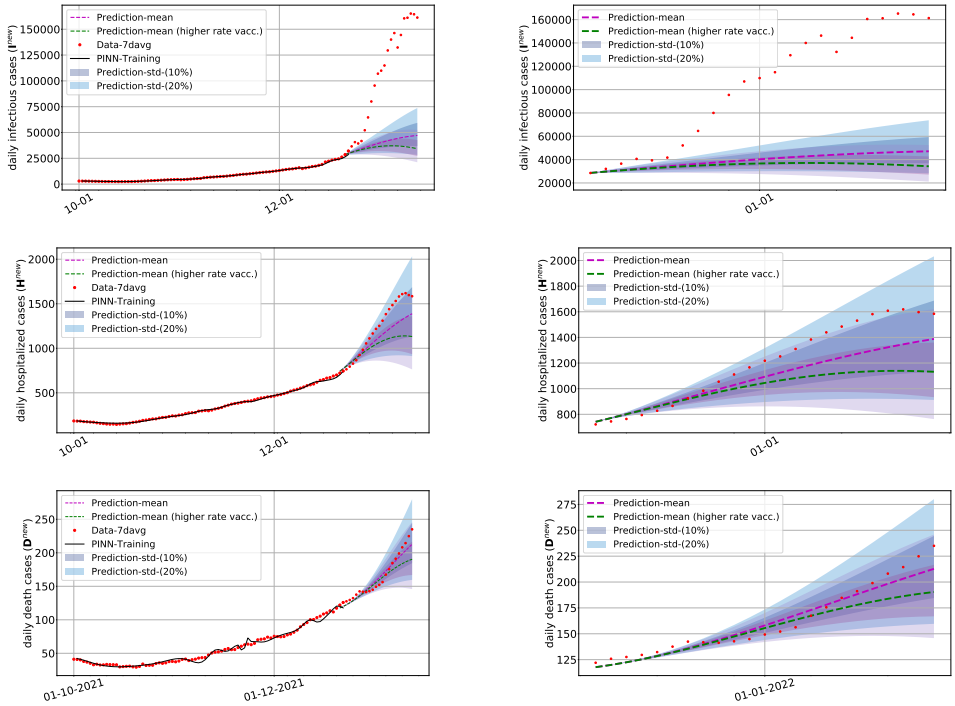

Figura 4.9: Nelle ascisse viene riportato il mese e il giorno (mm-gg). Nelle ordinate, invece, dall'alto verso il basso: il numero di nuovi infetti, ospedalizzati e deceduti giornalieri. A destra viene mostrato uno zoom sulla parte predittiva (dal 21/12/21 al 12/01/22)

Osservazione 16. Notiamo che:

- il numero dei nuovi infetti è sottostimato. Tuttavia, il periodo di previsione coincide con il diffondersi della variante Omicron in Italia, in cui c'è stata una crescita esponenziale del numero di nuovi casi nel mese di gennaio 2022.
- il numero di nuovi ospedalizzati e di nuovi deceduti differisce di "poco" ($<$ 500) rispetto alle previsioni.

Conclusioni

Gli obiettivi di questa tesi ruotavano attorno a tre aspetti: i dati disponibili sulla malattia COVID-19, i modelli matematici compartimentali che includessero le vaccinazioni e l'utilizzo di reti neurali "physics-informed", un nuovo approccio basato sul *deep learning* che mette insieme i primi due aspetti.

I tre aspetti sono stati dapprima approfonditi singolarmente nei primi tre capitoli di questo lavoro e si sono poi applicate le PINNs al modello SEIJDHR. Infine i risultati ottenuti sono stati riportati nel quarto capitolo ed in particoloare:

- sono stati effettuati test di tre codici Python su dati COVID-19 riguardanti la città di New York;
- l'analisi è stata estesa alla realtà italiana reperendo i dati italiani sul numero di positivi, ospedalizzati e deceduti giornalieri;
- nell'indagine della parte predittiva riguardante i dati italiani, si è individuato un punto critico legato alla funzione che modella la percentuale di ricoveri; sono stati quindi eseguiti numerosi esperimenti per il controllo di tali previsioni.

Ulteriori indagini sono ancora in evoluzione ed è consigliato un approfondimento dei codici utilizzati.

Appendice A

Modello per valutare la strategia di allocazione dei vaccini negli Stati Uniti

Presentiamo un recente modello per descrivere la dinamica dell'epidemia di COVID-19 negli Stati Uniti, utilizzato per valutare la performance della strategia di allocazione del vaccino adottata dal *Centers for Disease Control* (CDC) rispetto a quattro obiettivi: minimizzare la mortalità, i casi, le infezioni e gli anni di vita persi (YLL). La demografia della popolazione è stratificata in 17 sottopopolazioni per età, comorbidità, occupazione e condizione di vita (congestionata o non congestionata). La strategia di allocazione del CDC consiste in quattro gruppi di vaccinazione prioritari ed è paragonata a tutte le altre possibili allocazioni ottimali che scaglionano il lancio del vaccino in un massimo di quattro fasi.

Gli individui vengono distinti in:

- S, ovvero i suscettibili al virus;
- E, esposti, gli infettati di recente ma che non hanno ancora diffuso il virus;
- A, infetti ma asintomatici, ovvero coloro non hanno mai mostrato sintomi e diffondono il virus;
- RA, i guariti dopo un decorso asintomatico dell'infezione;
- P, pre-clinici, coloro che non mostrano ancora sintomi ma diffondono il virus;

- C, clinici, mostrano i sintomi e diffondono il virus;
- Q, quarantena, coloro che mostrano i sintomi ma non diffondono il virus a causa dell'isolamento o dell'ospedalizzazione;
- RC, guariti, coloro che non diffondono più il virus dopo aver avuto sintomi;
- D, i deceduti a causa del virus.

Inoltre, ogni individuo è vaccinato (V), disposto ad essere vaccinato (W) o non disposto ad essere vaccinato (N). Combinando le precedenti classificazioni degli invividui, $\forall\ i = 1, ..., 17$ (uno per ogni sottopopolazione), si ottengono 20 diversi compartimenti: S_i^N, S_i^W, S_i^V, E_i^N, E_i^W, E_i^V, A_i^N, A_i^W, A_i^V, RA_i^N, RA_i^W, RA_i^V, P_i^N, P_i^W, P_i^V, C_i, C_i^V, Q_i, R_i^C, D_i.
La transizione tra i compartimenti viene schematizzata in figura A.1 e il modello è descritto dal sistema di ODEs (A.1).

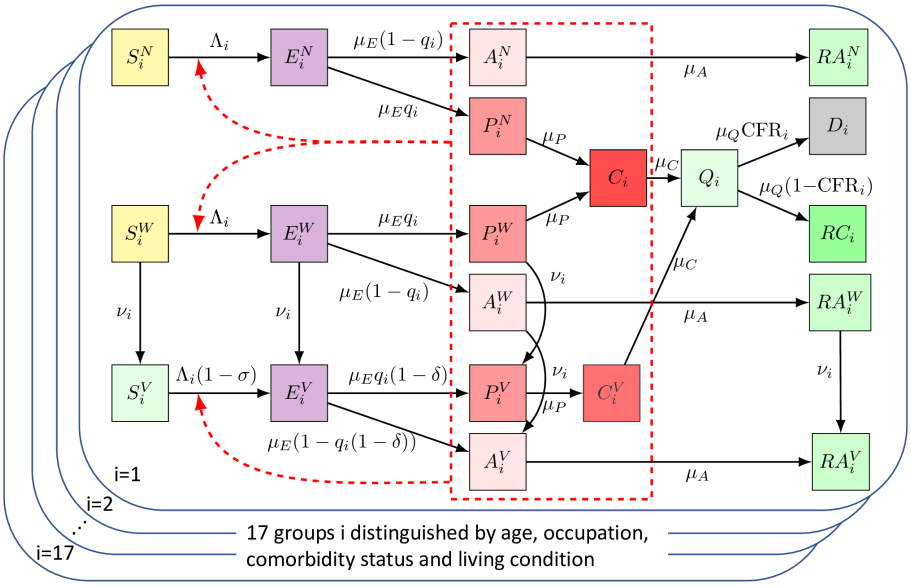

Figura A.1: Diagramma di trasferimento per il modello (A.1)
Dopo l'infezione, gli individui suscettibili (colonna più a sinistra) passano attraverso i vari compartimenti della malattia (colonne centrali) fino a raggiungere un compartimento finale (morte o recupero; colonne più a destra). Tutti gli individui pre-clinici, clinici e asintomatici possono causare nuove infezioni (casella rossa tratteggiata).
Fonte: [11]

$$
\begin{cases}
\dot{S_i^N} = -\Lambda_i S_i^N \\
\dot{S_i^W} = -\Lambda_i S_i^W - \nu_i S_i^W \\
\dot{S_i^V} = -\Lambda_i (1-\sigma) S_i^V + \nu_i S_i^W \\
\dot{E_i^N} = \Lambda_i S_i^N - \mu_E E_i^N \\
\dot{E_i^W} = \Lambda_i S_i^W - \mu_E E_i^W - \nu_i E_i^W \\
\dot{E_i^V} = \Lambda_i S_i^V (1-\sigma) - \mu_E E_i^V + \nu_i E_i^W \\
\dot{A_i^N} = \mu_E (1-q_i) E_i^N - \mu_A A_i^N \\
\dot{A_i^W} = \mu_E (1-q_i) E_i^W - \mu_A A_i^W - \nu_i A_i^W \\
\dot{A_i^V} = \mu_E (1-q_i(1-\delta)) E_i^V - \mu_A A_i^V + \nu_i A_i^W \\
\dot{RA_i^N} = \mu_A A_i^N \\
\dot{RA_i^W} = \mu_A A_i^W - \nu_i R A_i^W \\
\dot{RA_i^V} = \mu_A A_i^V + \nu_i R A_i^W \\
\dot{P_i^N} = \mu_E q_i E_i^N - \mu_P P_i^N \\
\dot{P_i^W} = \mu_E q_i E_i^W - \mu_P P_i^W - \nu_i P_i^W \\
\dot{P_i^V} = \mu_E q_i (1-\delta) E_i^V - \mu_P P_i^V + \nu_i P_i^W \\
\dot{C_i} = \mu_P (P_i^N + P_i^W) - \mu_C C_i \\
\dot{C_i^V} = \mu_P P_i^V - \mu_C C_i^V \\
\dot{Q_i} = \mu_C (C_i + C_i^V) - \mu_Q Q_i \\
\dot{RC_i} = (1 - \mathrm{CFR}_i) \mu_Q Q_i \\
\dot{D_i} = \mathrm{CFR}_i \mu_Q Q_i
\end{cases}
\tag{A.1}
$$

dove Λ_i rappresenta la forza di infezione per la sottopopolazione i, i = 1, . . . , 17 al tempo t ed è data da:

$$
\begin{aligned}
\Lambda_i = & \Phi(t)(1 - r(\text{casi attivi}))\beta_i \cdot \\
& \cdot \sum_{j=1}^{17} X_{ij} \frac{\left(f_A(A_j^N + A_j^W + f_V A_i^V) + P_j^N + P_j^W + f_V P_j^V + C_j + f_V C_j^V\right)}{N_j}
\end{aligned}
\tag{A.2}
$$

e ν_i rappresenta il tasso di vaccinazione degli individui disposti a ricevere il vaccino e senza una storia di infezione sintomatica da COVID-19. $\nu_i(t)$ dipende dalla strategia di assegnazione del vaccino e dal numero di vaccini disponibili giornalmente.

I parametri del modello vengono descritti nella tabella A.1.

I dati e i codici dello studio [11] sono reperibili dall'archivio `https://github.com/ckadelka/COVID19-CDC-allocation-evaluation`. Tuttavia,

PARAMETRO	DESCRIZIONE
q_i	frazione clinica dipendente dall'età
$1/\mu_E$	periodo di incubazione
$1/\mu_A$	tempo medio di diffusione del virus da parte di individui asintomatici
$1/\mu_P$	tempo medio di diffusione del virus prima della comparsa dei sintomi
$1/\mu_C$	tempo medio di diffusione del virus dopo la comparsa dei sintomi
$1/\mu_Q + \mu_C$	tempo medio tra l'insorgenza dei sintomi e la possibile morte
CFR_i	indice di fatalità dipendente dalla sotto-popolazione
σ e δ	riduzione delle infezioni e delle infezioni sintomatiche (quando infette) tra gli individui vaccinati (rispetto ai non vaccinati)
r(casi attivi)	livello generale di distanziamento sociale basato sul numero attuale di casi attivi
X_{ij}	numero medio giornaliero di contatti che un individuo della sottopopolazione i ha con individui della sotto-popolazione j
f_A	contagiosità relativa di individui asintomatici
f_V	contagiosità relativa degli individui vaccinati
N_i	numero di persone nella sottopopolazione i
β_i	suscettibilità all'infezione in base all'età

Tabella A.1: Descrizione dei parametri del modello (A.1)

si sono riscontrati problemi durante i test di tali codici, che non hanno consentito la validazione dei risultati; per questo motivo si è preferito ometterli.

Bibliografia

[1] *OMS (Organizzazione Mondiale della Sanità), 2020. L'OMS caratterizza la malattia COVID-19 come pandemia.* https://www.who.int/director-general/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020.

[2] *ONU (Nazioni Unite), 2020. L'impatto sociale del COVID-19.* https://www.un.org/development/desa/dspd/2020/04/social-impact-of-covid-19/.

[3] M. Raissi, P. Perdikaris e G.E. Karniadakis. "Physics-informed neural networks: A Deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *J. Comput. Phys. Processing* 378 (2019), 686–707.

[4] E. Kharazmi, M. Cai e X. Zheng et al. "Identifiability and predictability of integer- and fractional-order epidemiological models using physics-informed neural networks". In: *Nat Comput Sci* 1 (2021), 744–753.

[5] Ehsan Kharazmi et al. *Identifiability and predictability of integer- and fractional-order epidemiological models using physics-informed neural networks.* Ver. V.1.0.0. Ott. 2021. DOI: 10.5281/zenodo.5565308. URL: https://doi.org/10.5281/zenodo.5565308.

[6] Li Michael Y. *An introduction to mathematical modeling of infectious diseases.* Mathematics of Planet Earth,2. Springer,Cham, 2018.

[7] M.J. Keeling e P. Rohani. "Modeling infectious diseases in humans and animals." In: *Princeton university press* (2007).

[8] JA Metz O. Diekmann JA Heesterbeek. "On the definition and the computation of the basic reproduction ratio R0 in models for infectious diseases in heterogeneous populations." In: *J Math Biol.* 28(4) (1990), 365–382.

[9] G. Macdonald. "The analysis of equilibrium in malaria". In: *Tropical Diseases Bulletin* 49.9 (1952), pp. 813–29.

[10] Diego Caccavo. "Chinese and Italian covid-19 outbreaks can be correctly described by a modified SIRD model". In: *medRxiv* (2020).

[11] M. R. Islam et al. "Evaluation of the United States COVID-19 vaccine allocation strategy". In: *PLoS One 16* (11), e0259700 (2021).

[12] F. et altr. Brauer. *Mathematical Epidemiology*. Springer, Berlin, Heidelberg, 2008.

[13] S.H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. (2nd ed.)CRC Press., 2015.

[14] Alfredo Bellen e Marino Zennaro. *Numerical methods for delay differential equations*. Numerical Mathematics e Scientific Computation, Oxford University Press, USA, 2003.

[15] Ian Goodfellow, Yoshua Bengio e Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016.

[16] Y. LeCun, Y. Bengio e G. Hinton. "Deep learning". In: *Nature* 521 (2015), pp. 436–444.

[17] M.A. Nielsen. *Neural networks and deep learning (Vol. 2018)*. San Francisco, CA, USA: Determination press, 2015.

[18] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[19] Tom M Mitchell et al. "Machine learning. 1997". In: *Burr Ridge, IL: McGraw Hill* 45.37 (1997), pp. 870–877.

[20] F. Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain". In: *Psychological Review, 65* (6) (1958), 386–408.

[21] *Immagine*. `https://commons.wikimedia.org/wiki/File:Neuron_-_annotated.svg`.

[22] David E. Rumelhart, Geoffrey E. Hinton e Ronald J. Williams. "Learning Representations by Back-propagating Errors". In: *Nature* 323.6088 (1986), pp. 533–536.

[23] C.F. Higham e D.J. Higham. "Deep learning: An introduction for applied mathematicians". In: *SIAM Review* 61(4) (2019), pp. 860–891.

[24] G.P.H. Styan. "Hadamard products and multivariate statistical analysis". In: *Linear Algebra and its Applications* 6 (1973), pp. 217–240.

[25] Atilim Gunes Baydin et al. "Automatic differentiation in machine learning: a survey". In: *Journal of Marchine Learning Research* 18 (2018), pp. 1–43.

[26] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[27] *Git*. https://en.wikipedia.org/wiki/Git.

[28] *Archivio dati*. https://github.com/CSSEGISandData/COVID-19.

[29] *Archivio dati*. https://github.com/govex/COVID-19.

[30] *Funzione di pandas*. https://pandas.pydata.org/docs/reference/api/pandas.melt.html.

[31] *Report vaccini*. https://www.governo.it/it/cscovid19/report-vaccini/.

[32] *Archivio dati*. https://github.com/nychealth/coronavirus-data.

[33] *Varianti del SARS-CoV-2 (Ministero della salute)*. https://www.salute.gov.it/portale/p5_1_2.jsp?id=250&lingua=italiano.

[34] *Varianti del SARS-CoV-2 (WHO)*. https://www.who.int/en/activities/tracking-SARS-CoV-2-variants/.

[35] *GISAID*. https://www.gisaid.org.

[36] *Nextstrain*. https://nextstrain.org.

[37] A.Rambaut, E.C.O'Toole e Á. et al. "A dynamic nomenclature proposal for SARS-CoV-2 lineages to assist genomic epidemiology." In: *Nat Microbiol* 5 (2020), 1403–1407.

[38] S. Khare et altri. "GISAID's Role in Pandemic Response[J]". In: *China CDC Weekly* 3(49) (2021), pp. 1049–1051.

[39] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv e-prints* (set. 2016), arXiv:1609.04747.

[40] Jorge Nocedal e Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[41] Xavier Glorot e Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. A cura di Yee Whye Teh e Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. PMLR, 2010, pp. 249–256.

Ringraziamenti

Desidero esprimere i miei sentiti ringraziamenti alla prof.ssa Zama per avermi guidata in questo lavoro di ricerca con costanza, passione e dedizione. Ringrazio anche la prof.ssa Piccolomini per i numerosi consigli e spunti di approfondimento e il dott. Sebastiani per aver contribuito nei test dei codici e non essersi arreso quando questi non mostravano i risultati sperati.

Ringrazio la mia famiglia per il supporto costante di questi anni e per credere sempre in me. In particolare ringrazio mio padre per avermi trasmesso l'amore per la matematica, mia madre per i preziosi consigli e mia sorella, Ilaria, per la grinta che mi trasmette e per l'affetto incodizionato che mi dimostra.

Ringrazio gli amici che ho avuto la fortuna di incontrare durante questo percorso univeritario: Martina, Matteo ed Eloy, compagni speciali che hanno condiviso con me le gioie e le ansie di questi due anni; Caterina e Benedetta per tutte le volte che mi avete fatto sentire meno la mancanza di casa.
Grazie alle mie amiche di sempre, Irene e Valeria, per essermi state vicino nonostante la lontananza; siete un punto fisso nella mia vita.

Infine, un grazie dal profondo del cuore va a Luca per spronarmi ogni giorno a puntare più in alto e per far venire fuori la versione migliore di me.