Modellazione e calibrazione del traffico autostradale per la rete di Eindhoven

Freeway traffic modeling and calibration for the Eindhoven network

Relatore: Prof. Alessandro Giua

Supervisor: Prof. Bart De Schutter (DCSC TUDelft)

Tutor: Dr. Monique van den Berg (DCSC TUDelft)

Tesi di Laurea di:

Federica Lamon

Sommario

1. Introduzione e obiettivo della Tesi
2. Analisi del caso di studio
3. Modellazione della rete autostradale
4. Implementazione Matlab del modello
5. Calibrazione del modello
6. Validazione
7. Contributi, risultati e conclusioni

Introduzione

I problemi di congestione del traffico, dovuto al rapido incremento dei veicoli in una rete stradale inadeguata, necessitano di una politica di gestione della viabilità che può includere:

- Estensione della rete stradale.
- Incremento dell'efficienza della rete stradale con mezzi pubblici.
- Uso migliore dell'infrastrutture esistenti.

Gestione dinamica del traffico

- Lo scopo della **gestione dinamica del traffico** è aiutare a raggiungere uno pieno sfruttamento della capacità della rete stradale esistente, con una conseguente diminuzione dei tempi di viaggio, degli ingorghi e aumento della sicurezza.
- In questa tesi presteremo particolare attenzione alla rete autostradale e ai problemi tipici di questo sistema.

Controllo dinamico del traffico: ciclo chiuso

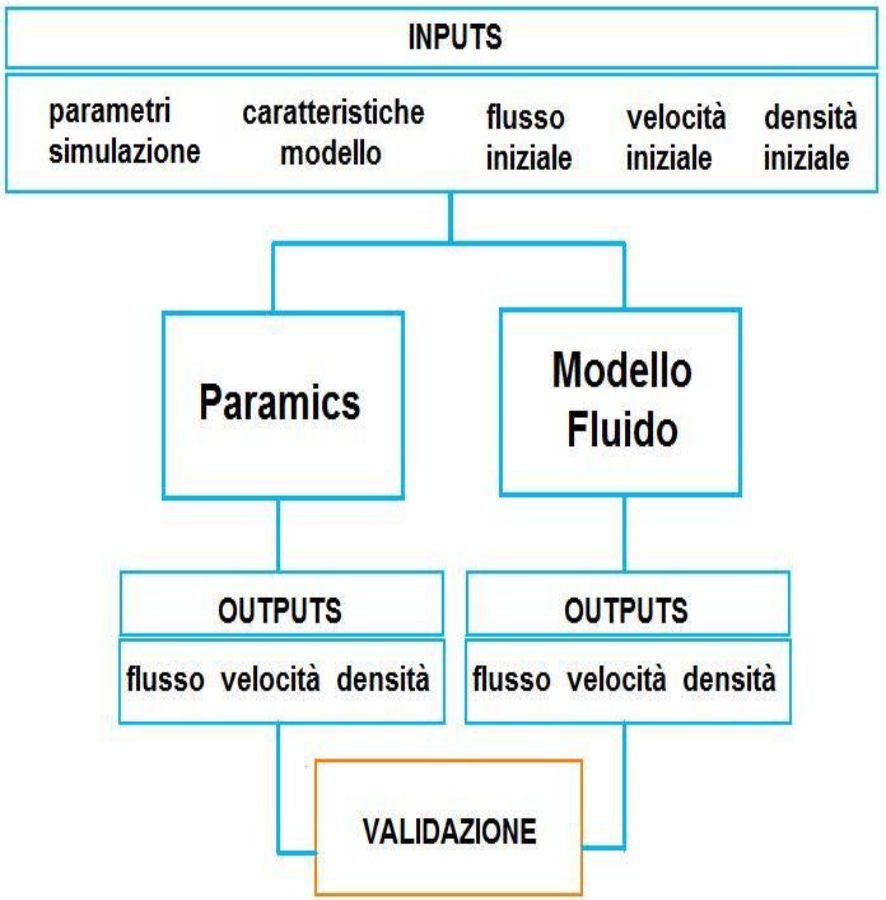
Model Predictive Control

- Model Predictive Control (MPC) è la tecnica di controllo utilizzata nell'ambito di questo progetto, che però non verrà approfondito in questa tesi. Essa appartiene alle tecniche di controllo ottimo che utilizzano un **modello interno di predizione (model-based).**
- L'MPC si basa su:

1. Lo stato corrente del traffico (flusso e velocità media di ogni tratta).
2. La domanda attesa del traffico.
3. Gli strumenti di controllo del traffico utilizzati e la loro collocazione nella rete.

Obiettivo della tesi

Scopo di questo lavoro è costruire il **modello interno di predizione,** utilizzato dall'MPC per calcolare la domanda attesa, e **validarlo**, ossia verificare che **le uscite del modello siano congruenti con i dati provenienti dal** software di simulazione di traffico Paramics Quadstone.



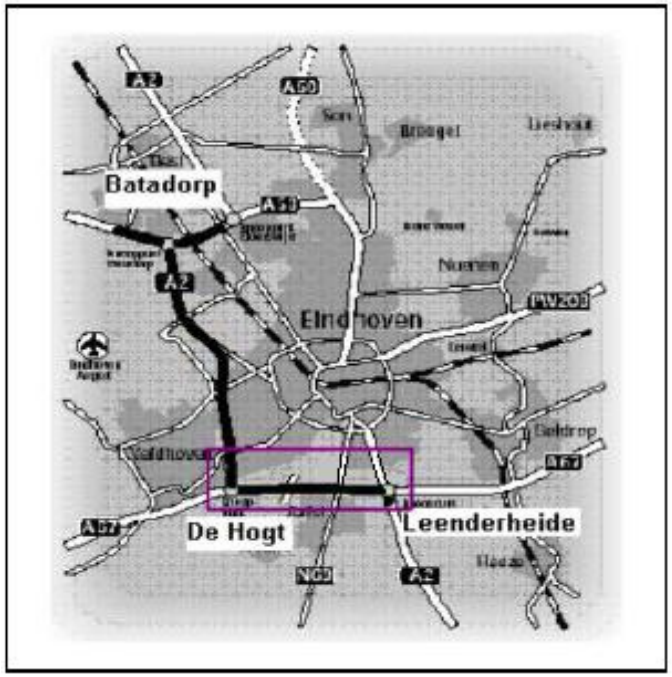
Paramics Quadstone

- E’ uno strumento di simulazione del traffico a **livello microscopico** ossia, i dati che fornisce descrivono il comportamento di ogni singolo veicolo.
- Le simulazioni effettuate con Paramics forniscono dati che rappresentano possibili scenari di traffico su cui testare il modello.
- E’ costituito da diversi tool, tra cui Paramics Modeller il quale fornisce una visualizzazione della rete stradale e della domanda del traffico utilizzando un’interfaccia grafica per l’utente (GUI).

Sommario

1. Introduzione e obiettivo della Tesi
2. Analisi del caso di studio
3. Modellazione della rete autostradale
4. Implementazione Matlab del modello
5. Calibrazione del modello
6. Validazione
7. Contributi, risultati e conclusioni

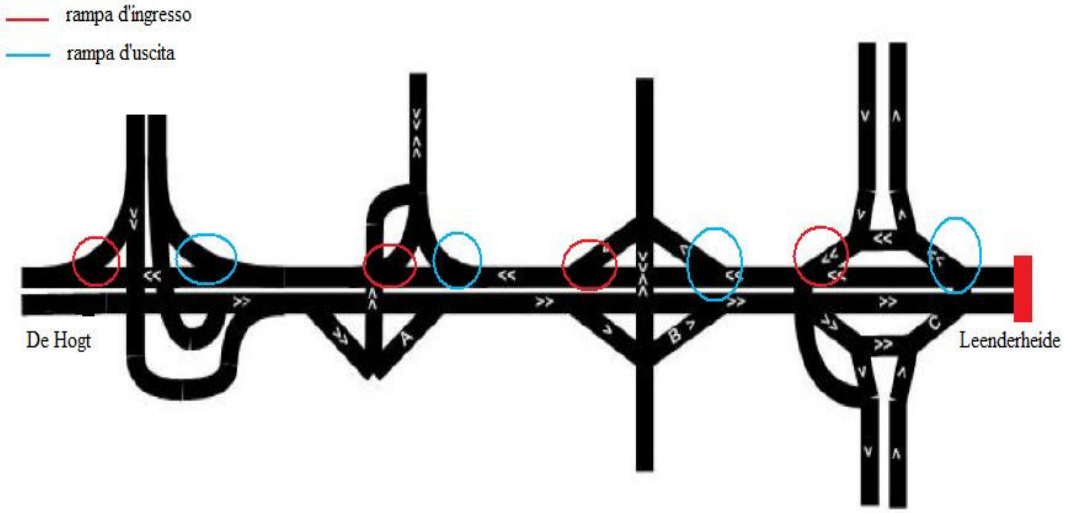
Eindhoven: caso di studio



- Il caso di studio riguarda un tratto di autostrada nei pressi di Eindhoven nella direzione Leenderheide-De Hogt lunga circa 6.5 km.

- La rete consiste:

1. Ingresso principale,
2. 4 rampe d'ingresso,
3. 4 rampe d'uscita.



Sommario

1. Introduzione e obiettivo della Tesi
2. Analisi del caso di studio
3. Modellazione della rete autostradale
4. Implementazione Matlab del modello
5. Calibrazione del modello
6. Validazione

Modellazione: METANET model

- **METANET** è il modello analitico utilizzato per descrivere la rete autostradale in analisi. Esso appartiene alla famiglia dei **modelli macroscopici**, in cui il traffico viene descritto in termini aggregati, ossia in termini di:

1. Densità di traffico (veh/km/lane)
2. Velocità media (km/h)
3. Flusso del traffico (veh/h)

Incongruenza di dati :

- Paramics, simulatore microscopico, gestisce dati puntuali validi veicolo per veicolo.
- Metanet, modello macroscopico, gestisce dati aggregati.

Soluzione: creazione di apposite funzioni Matlab che trasformano i dati validi puntualmente di Paramics, in dati aggregati, confrontabili con le uscite del modello.

Modellazione: METANET model

- METANET rappresenta, tramite un grafo orientato, la rete che viene suddivisa in:

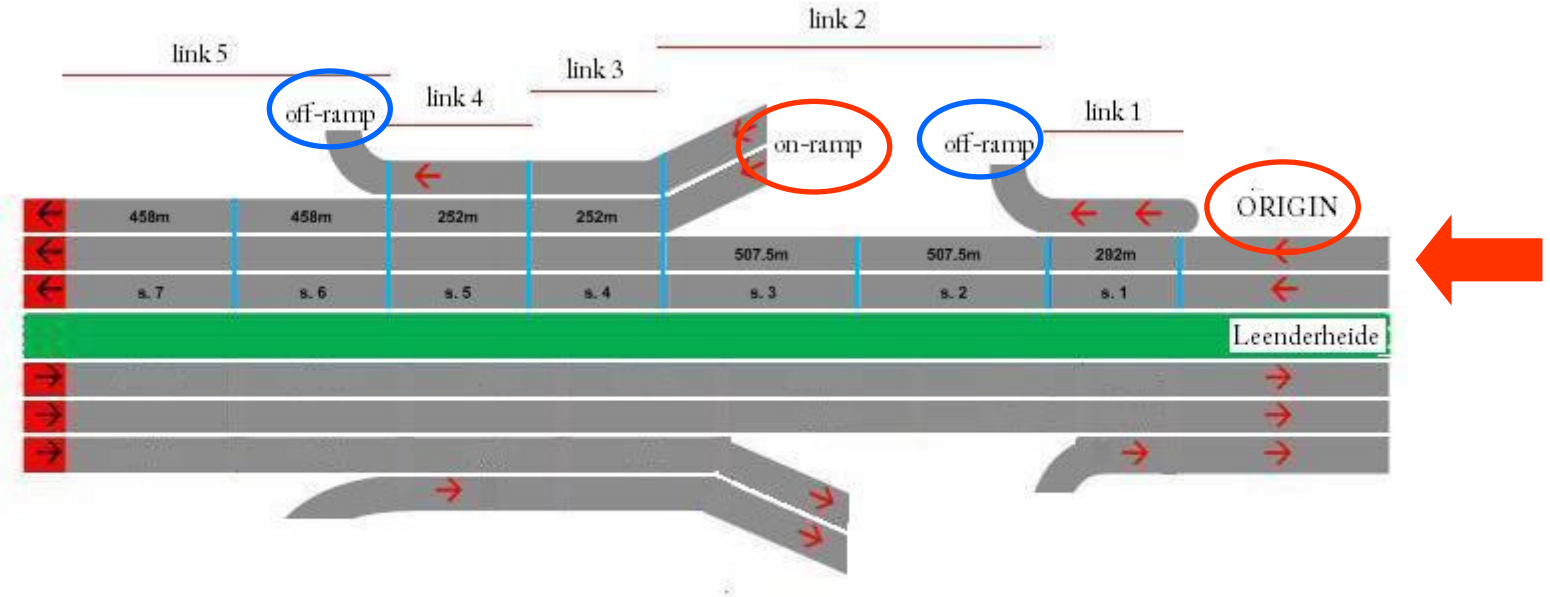
1. **Links** (indicati dall'indice m): individua tratti di autostrada con stesse caratteristiche.
2. **Segmenti** (indicati dall'indice i) : Ogni link viene suddiviso in N_m segmenti di pari lunghezza L_m.

Rampe d'ingresso e rampe d'uscita sono segmenti rappresentati da nodi.

Eindhoven: caso di studio

- La rete è stata suddivisa in 13 links, ognuno suddiviso in segmenti così come riportato in tabella.
- Ai links evidenziati in rosso si congiunge una rampa d'ingresso, a quelle in blu una rampa d'uscita.

link	Corsie /link	Segmento/link	Lungezza segmento (m)
1	3	1	292
2	2	2	507.5
3	4	1	252
4	4	1	252
5	3	2	458
6	3	2	326
7	4	1	196
8	3	2	330.5
9	3	2	390
10	3	1	256
11	4	1	74
12	2	2	367.5
13	2	1	436
totale	//	19	6.517 km



Schema della rete stradale analizzata: direzione Leenderheide-De Hogt (1°parte)

METANET: adattamento alla rete e implementazione in Matlab

- Ogni segmento è caratterizzato da tre variabili:

1. $\rho_{m,i}(k)$:Densità di traffico (veh/km/lane)
2. $v_{m,i}(k)$:Velocità media (km/h)
3. $q_{m,i}(k)$:Flusso in uscita (veh/h)

In cui **k** identifica il time step **T** dell'intervallo di simulazione.

Intervallo di simulazione = **10 minuti;**

Time step (T) = **10 secondi;**

Time steps totali: 60;

Le seguenti equazioni descrivono l'evoluzione nel tempo della rete.

Il flusso in uscita da ogni segmento è uguale alla densità di traffico moltiplicata per la velocità media e il numero di corsie di quel segmento

$$q_{m,i}(k) = \rho_{m,i}(k) v_{m,i}(k) \lambda_m$$

La densità di traffico di ciascun segmento al time step successivo (k+1) è dato da:

$$\rho_{m,i}(k+1) = \rho_{m,i}(k) + \frac{T}{L_m \lambda_m}(q_{m,i-1}(k) - q_{m,i}(k))$$

La velocità media al time step successivo (k+1) è data da:

$$v_{m,i}(k+1) = v_{m,i}(k) + \frac{T}{\tau}(V(\rho_{m,i}(k)) - v_{m,i}(k)) + \frac{T}{L_m} v_{m,i}(k)(v_{m,i-1}(k) - v_{m,i}(k))$$
$$- \frac{\eta T}{\tau L_m} \frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa} - \frac{\delta T q_0(k) v_{m,1}(k)}{L_m \lambda_m (\rho_{m,1}(k) + \kappa)}$$

Termine di rilassamento

$$V(\rho_{m,i}(k)) = v_{free,m} \exp\left[-\frac{1}{a_m} \left(\frac{\rho_{m,i}(k)}{\rho_{crit,m}} \right)^{a_m}, (1+\alpha) v_{control,m,i}(k) \right]$$

→ Velocità desiderata

La velocità media al time step successivo (k+1) è data da:

$$v_{m,i}(k+1) = v_{m,i}(k) + \frac{T}{\tau}(V(\rho_{m,i}(k)) - v_{m,i}(k)) + \frac{T}{L_m} v_{m,i}(k)(v_{m,i-1}(k) - v_{m,i}(k))$$
$$- \frac{\eta T}{\tau L_m} \frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa} - \frac{\delta T q_0(k) v_{m,1}(k)}{L_m \lambda_m (\rho_{m,1}(k) + \kappa)}$$

Termine di convezione

$$V(\rho_{m,i}(k)) = v_{free,m} \exp\left[-\frac{1}{a_m} \left(\frac{\rho_{m,i}(k)}{\rho_{crit,m}} \right)^{a_m}, (1+\alpha) v_{control,m,i}(k) \right]$$

→ Velocità desiderata

La velocità media al time step successivo (k+1) è data da:

$$v_{m,i}(k+1) = v_{m,i}(k) + \frac{T}{\tau}(V(\rho_{m,i}(k)) - v_{m,i}(k)) + \frac{T}{L_m} v_{m,i}(k)(v_{m,i-1}(k) - v_{m,i}(k))$$
$$- \frac{\eta T}{\tau L_m} \frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa} - \frac{\delta T q_0(k) v_{m,1}(k)}{L_m \lambda_m (\rho_{m,1}(k) + \kappa)}$$

Termine di anticipazione

$$V(\rho_{m,i}(k)) = v_{free,m} \exp\left[-\frac{1}{a_m} \left(\frac{\rho_{m,i}(k)}{\rho_{crit,m}} \right)^{a_m}, (1+\alpha) v_{control,m,i}(k) \right]$$

→ Velocità desiderata

La velocità media al time step successivo (k+1) è data da:

$$v_{m,i}(k+1) = v_{m,i}(k) + \frac{T}{\tau}(V(\rho_{m,i}(k)) - v_{m,i}(k)) + \frac{T}{L_m} v_{m,i}(k)(v_{m,i-1}(k) - v_{m,i}(k))$$
$$- \frac{\eta T}{\tau L_m} \frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa} - \frac{\delta T q_0(k) v_{m,1}(k)}{L_m \lambda_m (\rho_{m,1}(k) + \kappa)}$$

termine rampa d'ingresso

$$V(\rho_{m,i}(k)) = v_{free,m} \exp\left[-\frac{1}{a_m} \left(\frac{\rho_{m,i}(k)}{\rho_{crit,m}} \right)^{a_m}, (1+\alpha) v_{control,m,i}(k) \right]$$

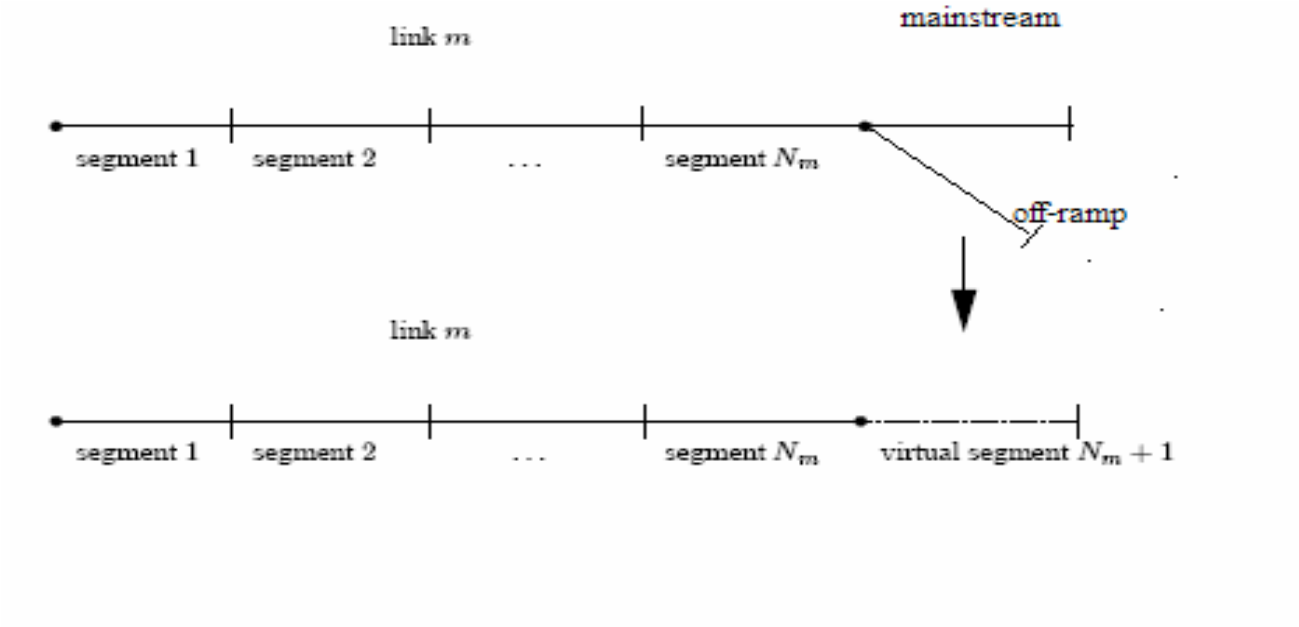
→ Velocità desiderata

Il flusso che entra nel nodo n, si suddivide in flusso che percorre la rampa d'uscita e flusso che rimane in autostrada. Quest'ultimo viene descritto da:

$$Q_n(k) = \sum_{\mu \in I_n} q_{\mu, N_\mu}(k)$$

$$q_{m,0} = \beta_{n,m}(k) \cdot Q_n(k)$$

Turning Rate



Sommario

1. Introduzione e obiettivo della Tesi
2. Analisi del caso di studio
3. Modellazione della rete autostradale
4. Implementazione Matlab del modello
5. Calibrazione del modello
6. Validazione
7. Contributi, risultati e conclusioni

Implementazione in Matlab

- La seguente funzione Matlab è stata creata per implementare il modello METANET opportunamente adattato alla rete di Eindhoven

- **`model_eindhoven`: interfaccia**

```
[Q_model,RHO_model,V_model]=model_eindhoven(v_free,a,rho_cr
   it,alpha,tau,eta,kappa,delta,vmin,vmax,rho_max)
```

Gli input della funzione sono i parametri descritti in precedenza, adattati al problema di ottimizzazione, selezionati per la successiva procedura di calibrazione, ossia:

`v_free`: [km/h] è la velocità media che gli automobilisti assumono se il traffico scorre agevolmente.

`a`: è usato come parametro per trovare la velocità desiderata.

`rho_crit`: [veh/km/corsia] è la densità critica per la quale il flusso del traffico è massimo. Un ingorgo potrebbe essere altamente probabile.

`alpha`: è coinvolto nel fattore di complicanza $(1+\alpha)$.

`tau`:[s] è il “tempo di rilassamento” ed è coinvolto sia nel termine di anticipazione che nel termine di rilassamento

`eta`: [km^2/h] è il “fattore di anticipazione” ed è coinvolto nel termine di anticipazione

`kappa`: [veh/km/corsia] parametro coinvolto nel termine di anticipazione

`delta`: rappresenta gli effetti relativi all’immissione di una rampa d’ingresso nella rete

`rho_max`: [veh/km/corsia] è la densità di traffico massima consentita

`v_min`: [km/h] è la velocità minima

`vmax`: [km/h] è la velocità massima

Gli output della funzione sono:

`Q_model`: Contiene i valori del flusso in uscita da ogni segmento per ogni time step. Dimensioni (19x60).

`RHO_model`: Contiene i valori di densità del traffico di ogni segmento per ogni time step. Dimensioni (19x60).

`V_model`: Contiene i valori di velocità media di ogni segmento per ogni time step. Dimensioni (19x60).

La funzione `model_eindhoven` richiama al suo interno le seguenti funzioni:

1. **`[q_0,q_ramp,rho_0,rho_ramp,V_0,v_ramp]=data_in();`**

- `q_0=` flusso di traffico in uscita dall'origine principale;
- `q_ramp=` flusso di traffico in uscita dalle rampe d'uscita;
- `rho_0=` densità di traffico all'origine principale;
- `rho_ramp=` densità di traffico in uscita alle rampe d'ingresso;
- `V_0=` velocità media del traffico all'origine principale;
- `v_ramp=` velocità media del traffico alle rampe d'ingresso.

che fornisce al modello i dati iniziali in ingresso.

2. **`beta=data_beta();`**

- `beta=` turning rate

che calcola il turning rate.

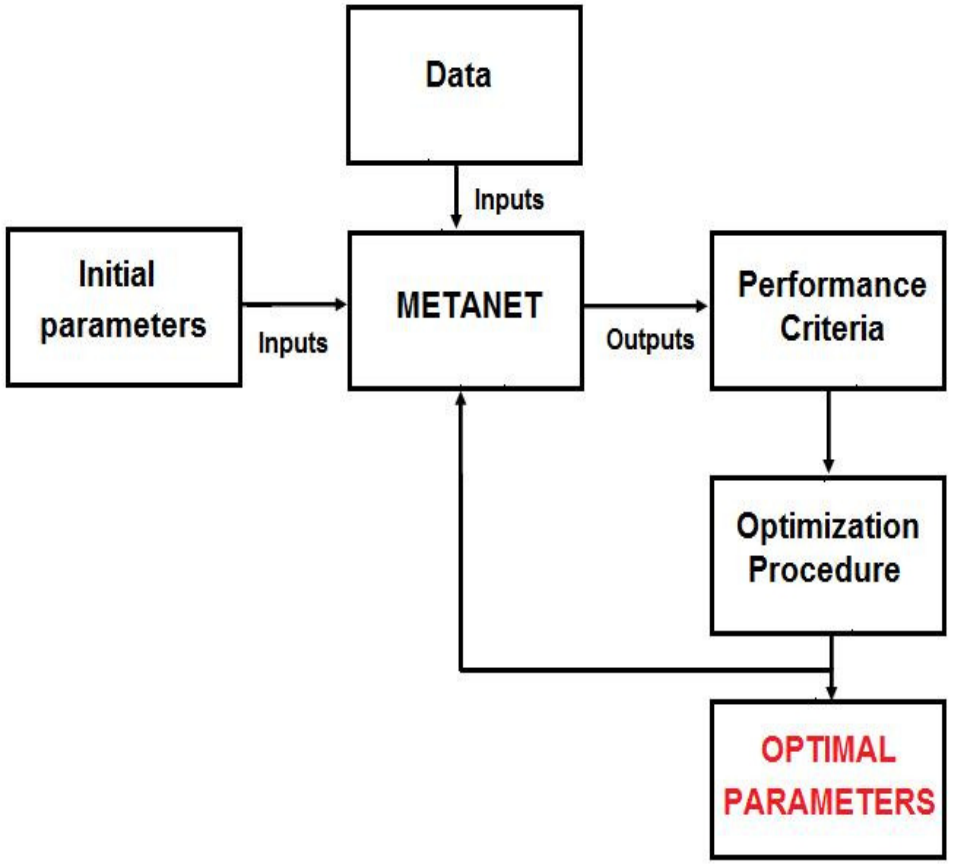
Queste funzioni sono state appositamente create per trasformare i dati provenienti dalle simulazioni eseguite con Paramics, che valgono puntualmente (microsimulazione), in dati aggregati utilizzabili dal modello macroscopico,

Sommario

1. Introduzione e obiettivo della Tesi
2. Analisi del caso di studio
3. Modellazione della rete autostradale
4. Implementazione Matlab del modello
5. Calibrazione del modello
6. Validazione
7. Contributi, risultati e conclusioni

Calibrazione del modello

- Lo scopo principale della calibrazione è dare una stima ottimale dei parametri del modello in modo che questo possa fornire delle uscite (flusso in uscita, densità di traffico, velocità media) in buona consistenza con i dati provenienti dalle simulazioni con Paramics.



Funzione obiettivo

- Con la **calibrazione** vengono trovati i valori dei parametri che minimizzano l'uscita della funzione obiettivo, ossia la differenza fra le uscite del modello e i dati di Paramics.

$$F = \sum_{k=1}^{60}\sum_{1=1}^{19}\left(\frac{Q_{\mathrm{mod}} - Q_{sim}}{Q_{average}}\right)^2 + \sum_{k=1}^{60}\sum_{i=1}^{19}\left(\frac{V_{\mathrm{mod}} - V_{sim}}{V_{average}}\right)^2$$

In cui Q_{mod} e V_{mod}, sono le matrici (19x60) che contengono le uscite del modello, mentre Q_{sim} e V_{sim} sono le matrici (19X60) che contengo i dati provenienti dalla simulazione con Paramics, opportunamente elaborati dalla funzione:

```
[Q_sim,RHO_sim,V_sim]=data_sim();
```

Calibrazione: funzioni di minimizzazione

Sono stati testati due tools di ottimizzazione per il problema di minimizzazione esposto.

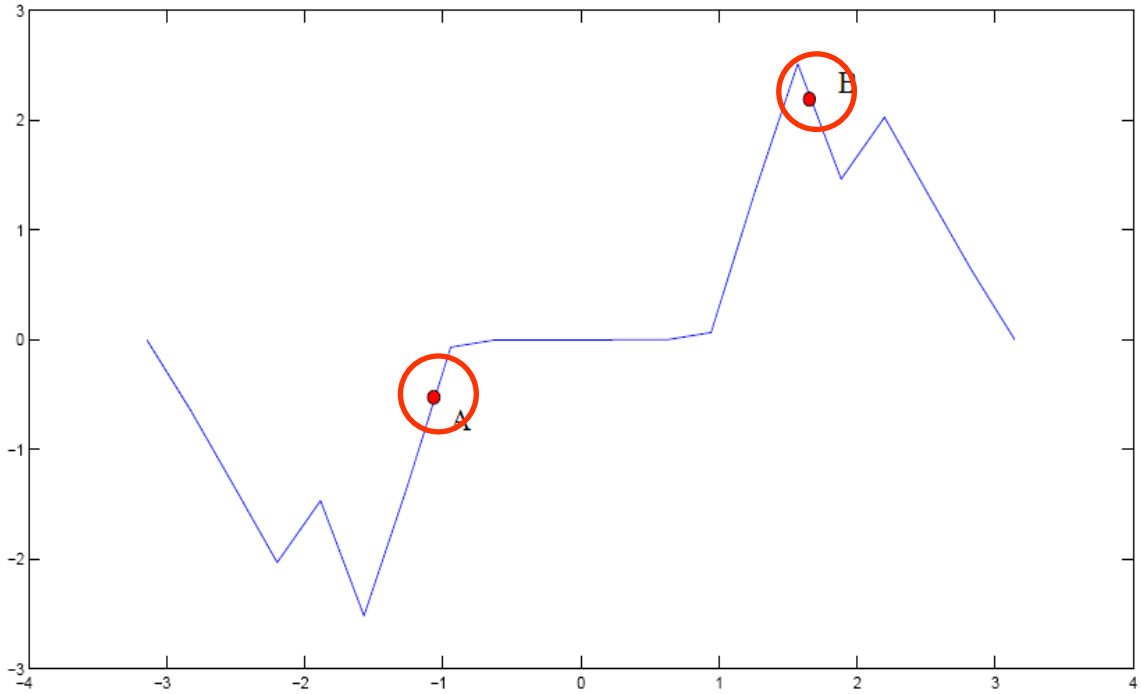
`fmincon`: metodo di ottimizzazione basato sul gradiente (gradient-based optimization method)- La ricerca dell'ottimo si ottiene seguendo la direzione del gradiente della funzione obiettivo.

`patternsearch`: metodo di ottimizzazione indipendente dal gradiente della funzione obiettivo (direct-search optimization method).

fmincon

- `fmincon` risulta un metodo non soddisfacente in quanto, essendo basato sul gradiente, non è adatto a funzioni non differenziabili o che possiedono molti minimi locali.

`fmincon`: a partire da un punto iniziale , fornito dall'utente, fmincon segue la pendenza della funzione per trovarne il minimo.



Funzione esempio

Patternsearch

Patternsearch utilizza un algoritmo iterativo per la ricerca del minimo della funzione obiettivo.

1. Ad ogni iterazione l'algortmo cerca tra un insieme di punti (mesh) dell'intorno del punto corrente, un punto che possa diminuire il valore della funzione obiettivo.
2. Se l'algoritmo trova tale punto, questo diventa il punto corrente della successiva iterazione.
3. L'algoritmo si ferma quando non trova più un punto che possa diminuire la funzione obiettivo.

Interfaccia matlab:

```
[x,z,exitflag,output]=patternsearch(@(x)obj_func(x),
    x0,[],[],[],[],lb,ub,[],option)
```

Inputs:

1. **`(x)obj_func(x)`:** funzione obiettivo,
2. **`x0`:** punto iniziale,
3. **`lb`:** limite superiore per **`x`**,
4. **`ub`:** limite inferiore per **`x`**,
5. **`option`:** opzioni varie per l'algoritmo.

Outputs:

1. **`x`:** valore del punto di minimo,
2. **`z`:** valore della funzione obiettivo nel punto di minimo,
3. **`exitflag`:** stato dell'algoritmo all'ultima iterazione,
4. **`output`:** informazioni varie sull'algoritmo.

Risultati raggiunti

Tramite l'utilizzo di patternsearch, a partire dal seguente punto iniziale, scelto fra un insieme di punti generati casualmente:

```
x0=[115,1,30,0.5,13.5,55,45,0.5,5,145,70]
```

E' stato trovato il seguente punto di minimo:

```
x=[101,2,50,0.5,11.55,30,20,0.51,10,156,70]
```

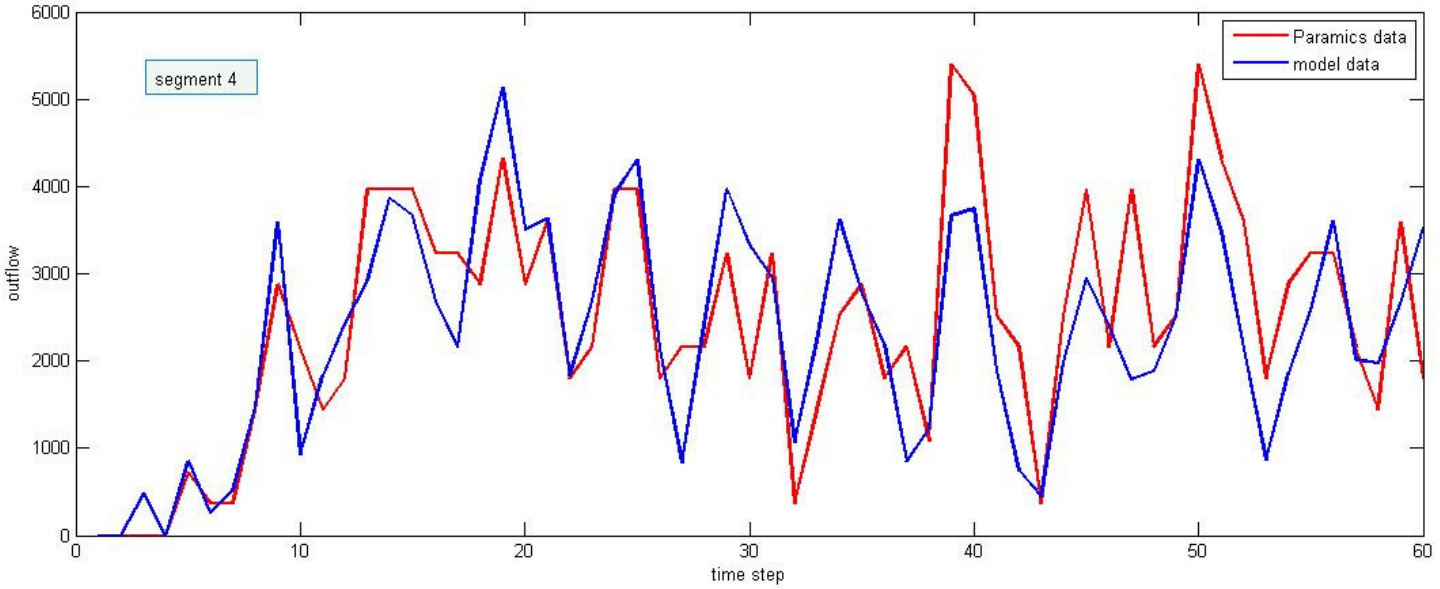
v_free	101
a	2
rho_crit	50
alpha	0.5
eta	11.55
tau	30
kappa	20
delta	0.51
vmin	10
vmax	156
rho_max	70

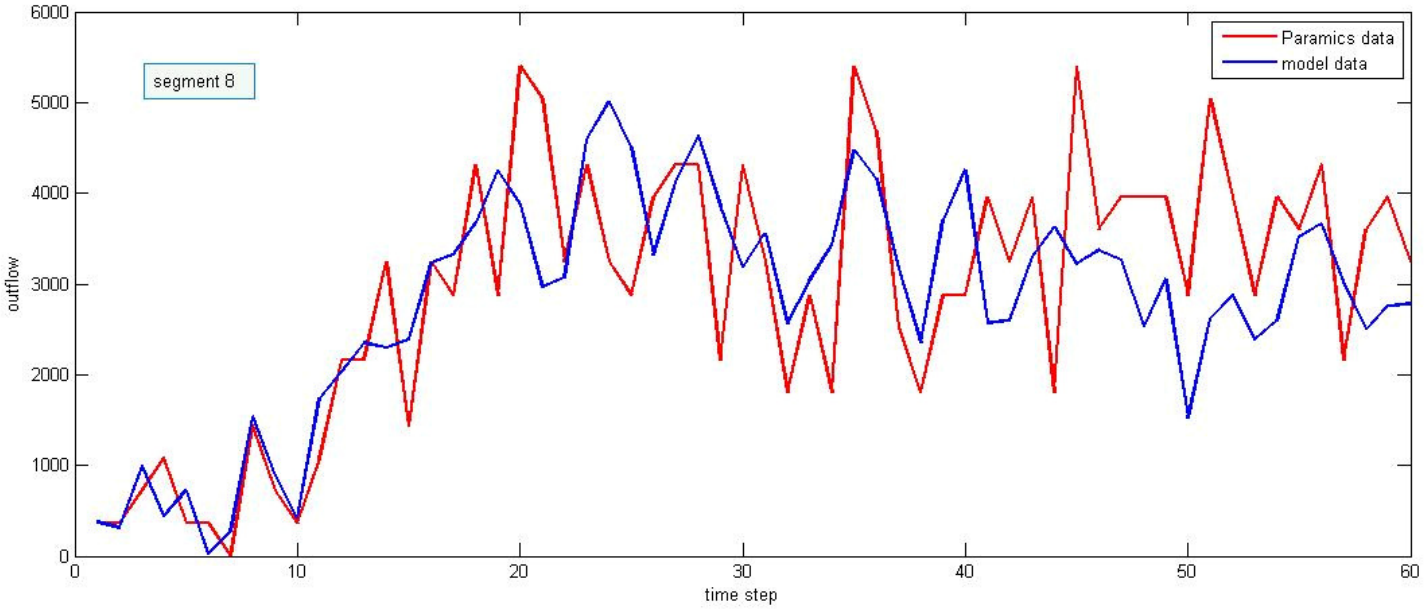
Sommario

1. Introduzione e obiettivo della Tesi
2. Analisi del caso di studio
3. Modellazione della rete autostradale
4. Implementazione Matlab del modello
5. Calibrazione del modello
6. Validazione
7. Contributi, risultati e conclusioni

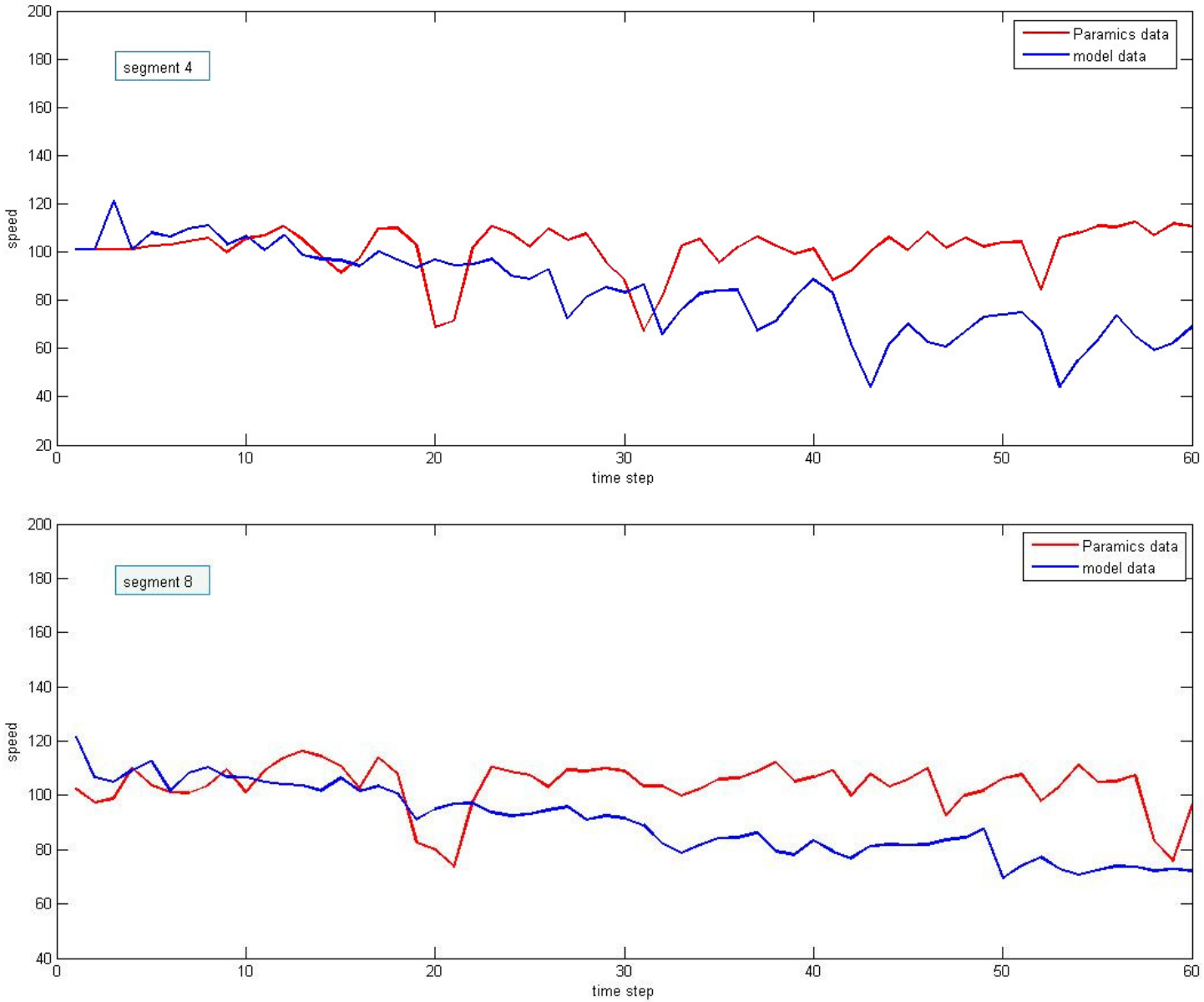
Validazione

- Col termine validazione si intende la verifica dei risultati raggiunti.
- Il comportamento del modello viene testato su un altro set di dati di simulazione di Paramics diversi da quelli con cui il modello è stato calibrato, in modo da verificare che questo si comporti in modo soddisfacente anche sotto altre condizioni.

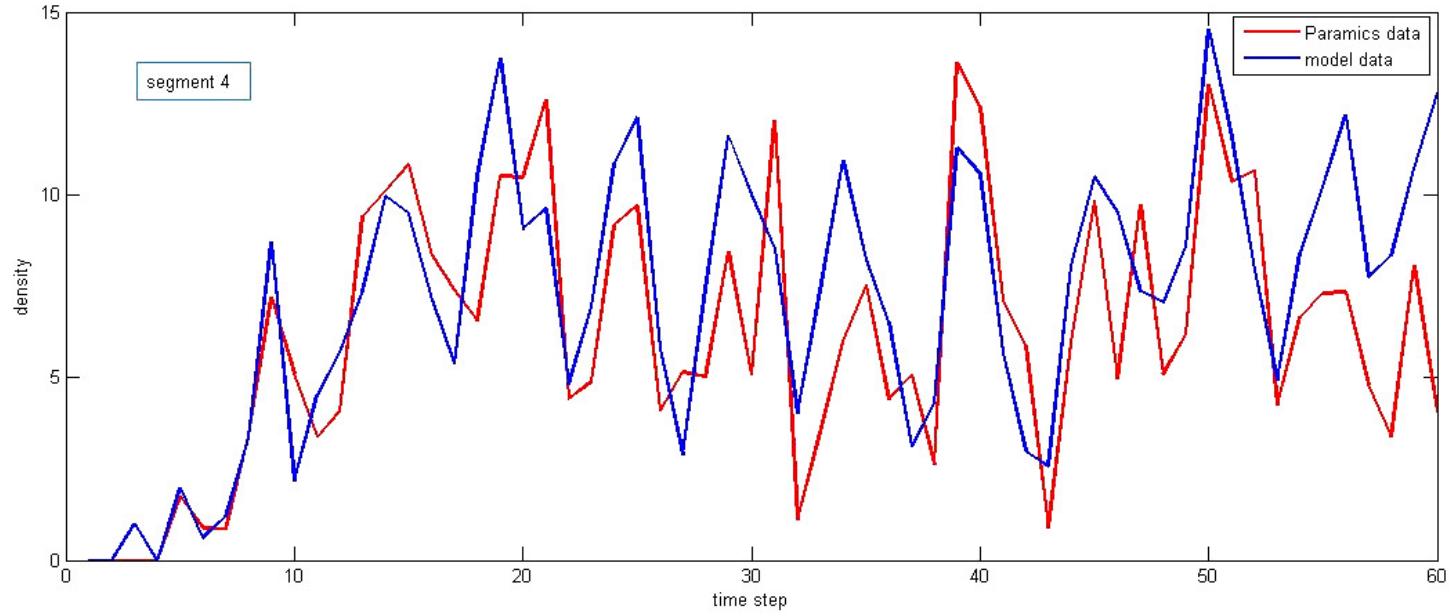


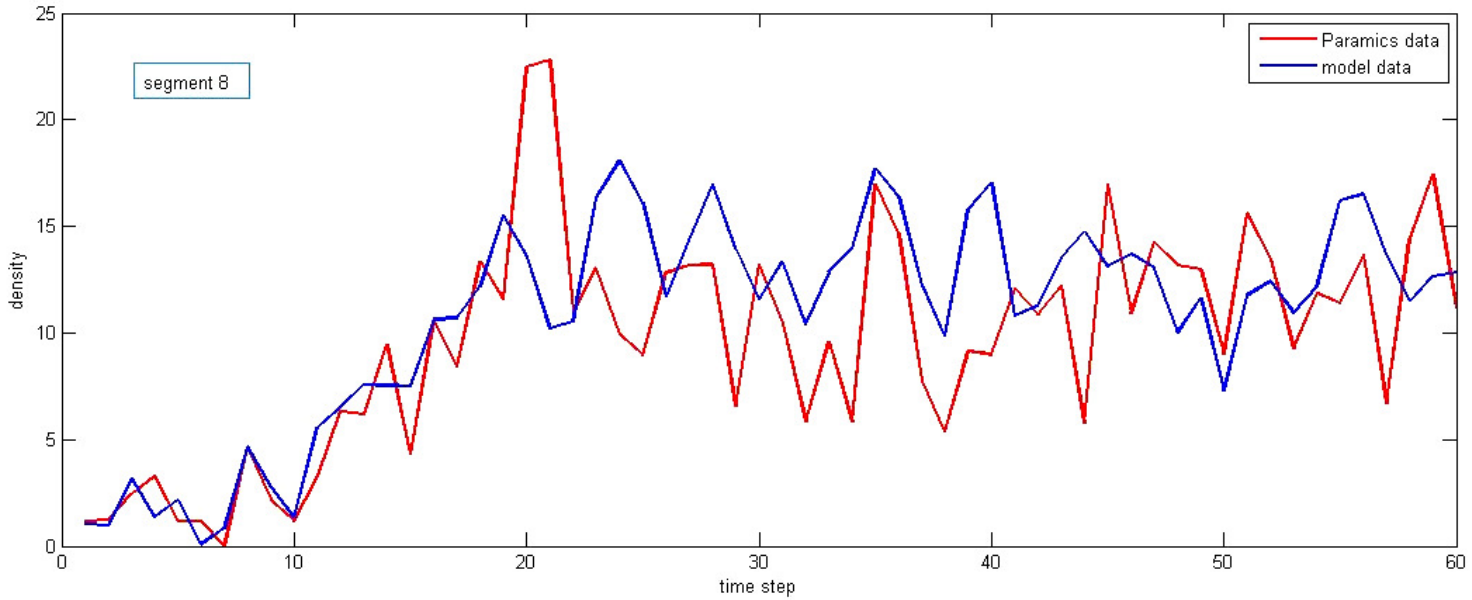


Confronto fra valori di flusso per il 4° e 8° segmento



Confronto fra valori di velocità per il 4° e 8° segmento





Confronto fra valori di densità per il 4° e 8° segmento

Valore della funzione obiettivo

I grafici visti presentano un buon andamento del modello, tuttavia per alcuni segmenti i risultati non sono altrettanto buoni, tanto che l'errore medio calcolato fra tutti i segmenti risulta:

- Errore medio relativo al flusso di traffico: 60%
- Errore medio relativo alla densità di traffico: 55%
- Errore medio relativo alla velocità: 12%

Il segmento 16 introduce un errore consistente nella valutazione globale.

E' un segmento problematico perché molto corto, e quindi difficilmente trattabile dal modello METANET che introduce grandezze medie che meglio si adattano a segmenti più lunghi.

Il segmento 16 inoltre introduce un errore anche sui segmenti successivi.

Valutando l'errore solo per i primi 15 segmenti otteniamo:

Errore flusso=28%

Errore densità=32%

Errore velocità=11%

segmento	Errore medio flusso	Errore medio densità	Errore medio velocità
1	0.2819	0.2933	0.2267
2	0.2787	0.2439	0.2867
3	0.2234	0.2611	0.3528
4	0.2129	0.2461	0.1053
5	0.3082	0.3655	0.0996
6	0.3358	0.3703	0.1084
7	0.3498	0.364	0.0888
8	0.2333	0.2582	0.0452
9	0.3105	0.3227	0.0467
10	0.2537	0.2576	0.0505
11	0.3713	0.3896	0.0527
12	0.2793	0.2903	0.0464
13	0.2733	0.2875	0.0429
14	0.3151	0.3252	0.0341
15	0.2994	0.3339	0.0225
16	2.3152	2.2216	0.2975
17	1.9013	1.9904	0.1456
18	1.9463	1.1627	0.1142
19	1.1278	1.1754	0.1466

errori medi nel tempo per segmento.

Sommario

1. Introduzione e obiettivo della Tesi
2. Analisi del caso di studio
3. Modellazione della rete autostradale
4. Implementazione Matlab del modello
5. Calibrazione del modello
6. Validazione
7. Contributi, risultati e conclusioni

Contributi

- Modellazione, calibrazione e validazione di una rete reale.
- Test del modello METANET su un caso reale.
- Sviluppo di una serie di interfacce Matlab per aggregare i dati delle simulazioni microscopiche confrontabili con le uscite macroscopiche del modello.

Risultati

- La descrizione aggregata del modello Metanet risulta accettabile soprattutto per la velocità media, di meno per flusso e densità.
- Questo ha potuto però evidenziare una difficoltà di rappresentazione da parte del modello METANET di segmenti molto corti.

Conclusioni

- Le differenze tra le uscite del modello e i dati della simulazione possono comunque essere migliorate adottando importanti accorgimenti:

1. Diminuire la durata del time step, in modo da ottenere risultati più accurati
2. Ottimizzazione dei parametri differenziati per ogni segmento.

GRAZIE PER L’ATTENZIONE!