Optimal perimeter control for two urban regions with macroscopic fundamental diagrams: A model predictive approach

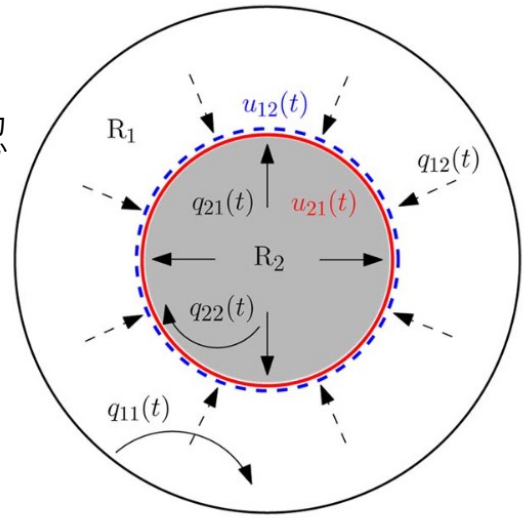
Geroliminis, N., Haddad, J., & Ramezani, M. (2012).
IEEE Transactions on Intelligent Transportation Systems, *14*(1), 348-359.

2022年4月26日

修士2年　増田 慧樹

Abstract

- ✓ **Macroscopic Fundamental Diagram (MFD)** を利用して，**2地域間交通の最適制御モデル**を定式化.
- ✓ 目的地へ到達するトリップ数が最大になるように2地域の境界で流出入を制御する.
- ✓ 最適制御を**モデル予測制御 (MPC: Model Predictive Control)** によって求める.
- ✓ 様々な混雑レベルに適用し，MFDのバラつきと需要のノイズに対して頑健性を確認
- ✓ 滑らかな制御を行うために2つの方法を提案.
- ✓ MPCが”貪欲な”フィードバック制御に比べて著しく良い性能を示すことを確認



構成

1. Introduction
2. Two-region macroscopic fundamental diagrams system
3. Model predictive control for the two-region macroscopic fundamental diagrams problem
   A. Two-region MFDs prediction model and optimization problem
   B. Two-region MFDs plant
   C. Greedy controller (GC)
   D. Tuning the prediction and control horizon parameters
4. Case study examples
5. Smoothing control
   A. Constraints for smoothing control
   B. Modified objective function
6. Discussion

大規模都市ネットワークにおける交通マネジメント

問題意識：大規模都市ネットワークにおける効率的な観測と交通マネジメント

1. 微視的 (microscopic) なアプローチ
   - 大規模NWの全リンク・信号交差点の交通をモデル化するのは複雑
   - もしモデル化できたとしても，中央制御は困難（計算負荷・利用者の選択の変化）

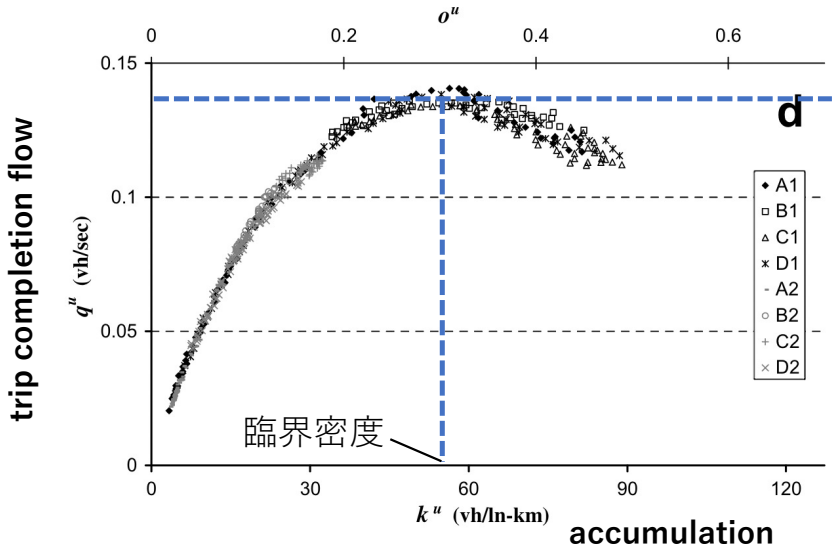
2. 巨視的 (macroscopic) なアプローチ

- **Macroscopic Fundamental Diagram (MFD)**は，巨視的な指標により混雑現象を捉えられる
  → シンプル！
- MFDを用いたエレガントな制御法を提案する.

Macroscopic Fundamental Diagram (MFD)とは

ネットワーク内の車両存在台数 (/平均交通密度) とトリップ完了率 (/平均交通量) の関係

accumulation 観測可能　　**trip completion flow** 観測困難



→観測可能な平均交通量とほぼ線形の関係であることを実データから確認（Geroliminis and Daganzo, 2008）

横浜のMFD（Geroliminis and Daganzo, 2008）

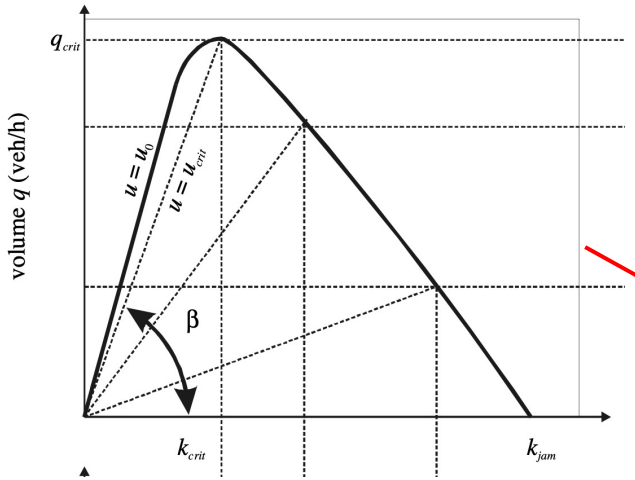
- ネットワークが均質 (homogeneous) である必要がある
  ＝リンク密度の空間的な分散が小さいということ
- MFDの形はOD需要の変動の影響をあまり受けない→OD表を詳細に調べなくていい

◎ 地域間の境界で地域内の密度を操作する (perimeter control) だけで渋滞を抑制できるかも！

Macroscopic Fundamental Diagram (MFD)とは

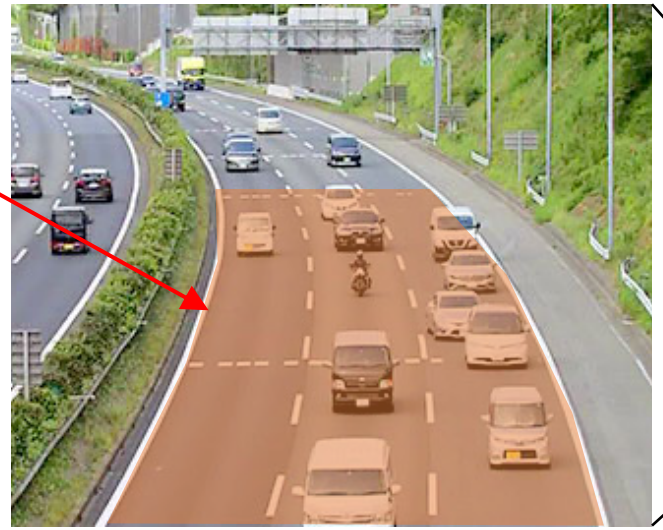
cf.) fundamental Diagram

均質な道路区間の交通量・密度・速度の関係



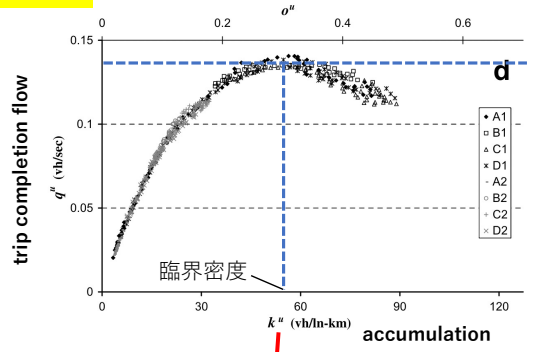
q-k図

ミクロ



マクロ

MFD



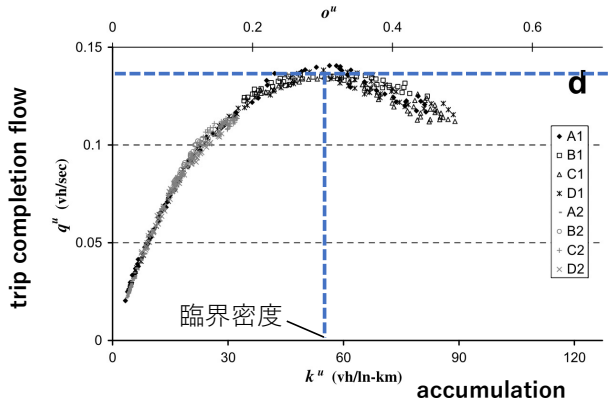
課題の整理

- 1地域のMFDの最適制御の研究はある (Daganzo, 2007).

  車両存在台数が臨界密度付近になるように，流入交通量を制御する方針

  ↔ 複数地域の制御はより複雑

- MFDの誤差や需要のノイズに対応するためには，closed-loopな制御が必要
  →**モデル予測制御 (MPC: Model Predictive Control)** が使える！



【用語】

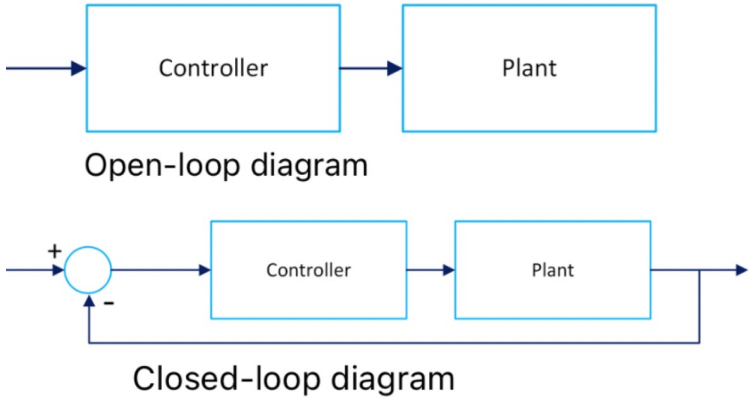
- open-loop制御

  事前に目的通りの挙動を行わせるための入力を算出して順次入力する

  →外乱に対処不能

- closed-loop制御（feedback制御）

  各時刻の出力及び状態と，外部から与えられる目標値を一致させるように制御する.

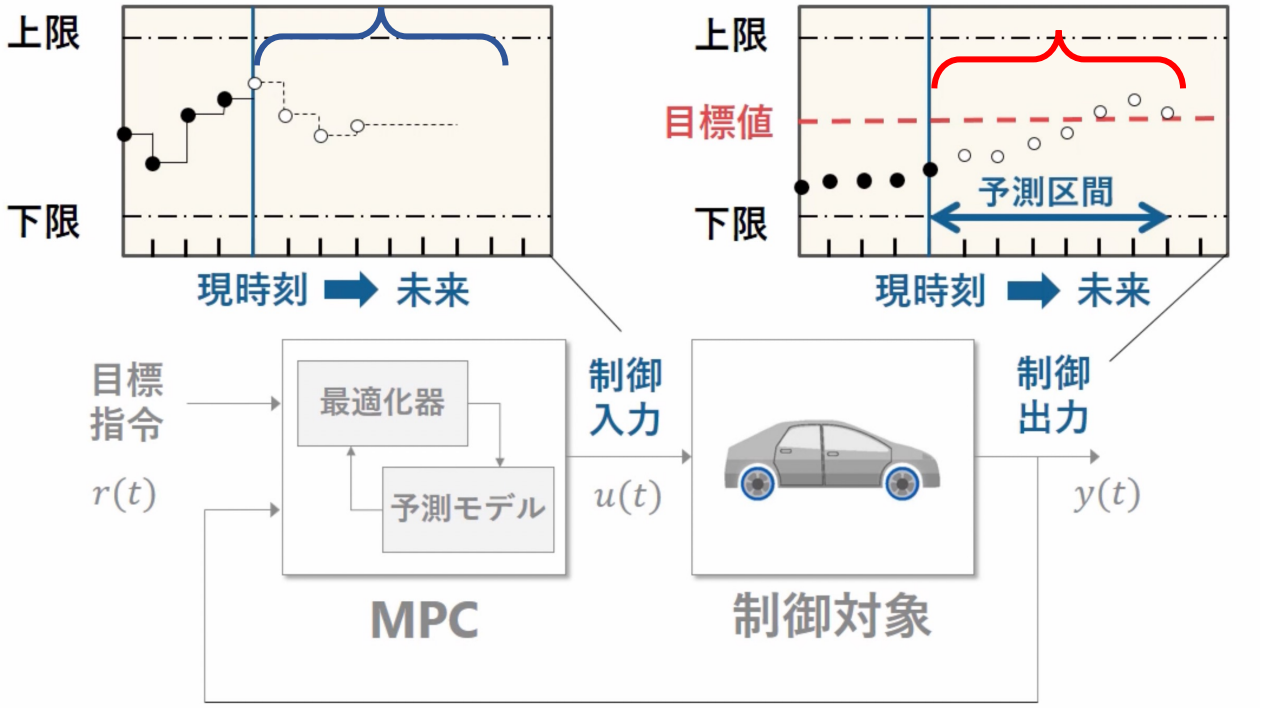


Model Predictive Control (MPC) とは

モデルによる予測と**オンラインの最適化**が特徴の制御手法

制御目的に応じた最適化問題を解いて制御入力を決定

数理モデルを利用して制御対象の**未来の挙動を予測**



- 汎用性が高い
- 各時刻の出力に対して予測と最適化を繰り返すフィードバック制御なので，誤差やノイズに強い
- 非線形最適化に帰着するので，制約条件を入れやすい

適用先：
高速道路ランプの流入制御・制限速度やルート案内情報の操作・信号制御など

https://jp.mathworks.com/videos/introduction-to-model-predictive-control-mpc-part-1-1494946684184.html

論文の流れ

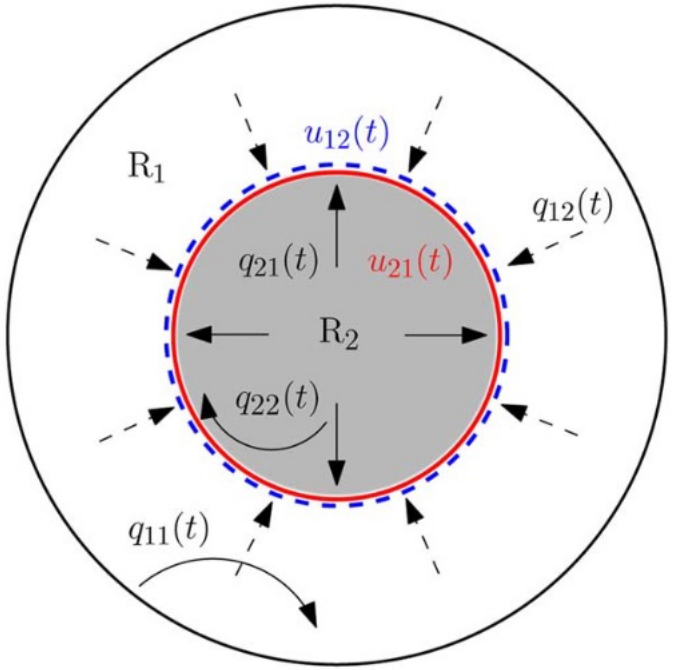
2章　MFDによって微分方程式モデルを作り，2地域の最適流出入制御 (perimeter control) を定式化

3章　MPCを定式化，パラメータのチューニング，比較用にGreedy Controller (GC) の導入

4章　MPCとGCのパフォーマンスを比較

5章　制御系列を滑らかにする2種の方法を提案

設定・notation



- 均質なNWからなる2地域 R_1, R_2 を考える
- $q_{11}(t), q_{22}(t)$：R_1, R_2の域内交通需要，　$q_{12}(t), q_{21}(t)$：R_1, R_2の域外交通需要
- $n_{ij}(t)$：時刻 t に R_i 内で R_j を目的地とする車両台数
- $n_i(t)$：時刻 t における R_i 内の総車両台数
- $G_i\left(n_i(t)\right)$：総車両台数 $n_i(t)$ に対するトリップ完了率（=MFD）
  = 三次関数で近似 $G_i\left(n_i(t)\right) = \mathrm{a_i n_i(t)^3 + b_i n_i(t)^2} + c_i n_i(t)$
- $i \to j$の流出交通量$M_{ij}(t) = \frac{n_{ij}(t)}{n_i(t)} \cdot G_i\left(n_i(t)\right), i \neq j$
- $i \to i$の域内交通量$M_{ii}(t) = \frac{n_{ii}(t)}{n_i(t)} \cdot G_i\left(n_i(t)\right)$
- 制御変数$u_{12}(t), u_{21}(t)$：流出率（混雑を防ぐため地域間の流出入交通を制限する）．1なら全ての車を流出させ，0なら全く流出させない．

2地域MFD制御問題の定式化

目的関数（評価関数）

終端時刻

$$J = \max_{u_{12}(t),u_{21}(t)} \int_{t_0}^{t_f} [M_{11}(t) + M_{22}(t)]\, dt$$

地域1→1で目的地に到着するフロー

目的地に到着した車両台数（トリップ完了した台数）を最大にするような制御入力の系列$u_{12}(t), u_{21}(t)$を求める

subject to

n_{ij}の保存則

$$\begin{cases} \dfrac{dn_{11}(t)}{dt} = q_{11}(t) + u_{21}(t) \cdot M_{21}(t) - M_{11}(t) & (2) \\ \dfrac{dn_{12}(t)}{dt} = q_{12}(t) - u_{12}(t) \cdot M_{12}(t) & (3) \\ \dfrac{dn_{21}(t)}{dt} = q_{21}(t) - u_{21}(t) \cdot M_{21}(t) & (4) \\ \dfrac{dn_{22}(t)}{dt} = q_{22}(t) + u_{12}(t) \cdot M_{12}(t) - M_{22}(t) & (5) \end{cases}$$

(3): demand　流出率　1→2のフロー

(5): demand　流入率　1→2のフロー　トリップ完了した車

$$0 \le n_{11}(t) + n_{12}(t) \quad (6)$$
$$0 \le n_{21}(t) + n_{22}(t) \quad (7)$$

各地域の台数は非負

$$n_{11}(t) + n_{12}(t) \le n_{1,\mathrm{jam}} \quad (8)$$
$$n_{21}(t) + n_{22}(t) \le n_{2,\mathrm{jam}} \quad (9)$$

各地域の台数に上限

$$u_{\min} \le u_{12}(t) \le u_{\max} \quad (10)$$
$$u_{\min} \le u_{21}(t) \le u_{\max} \quad (11)$$

流出入率の制御に上下限

$$n_{11}(t_0) = n_{11,0}\,;\, n_{12}(t_0) = n_{12,0}$$
$$n_{21}(t_0) = n_{21,0}\,;\, n_{22}(t_0) = n_{22,0} \quad (12)$$

初期状態

決定変数の$u(t)$は時間変化かつ連続 → 解法は，直接法，間接法，動的計画法などがある．

論文の流れ

2章　MFDによって微分方程式モデルを作り，2地域の最適流出入制御 (perimeter control) を定式化

3章　MPCを定式化，パラメータのチューニング，比較用にGreedy Controller (GC) の導入

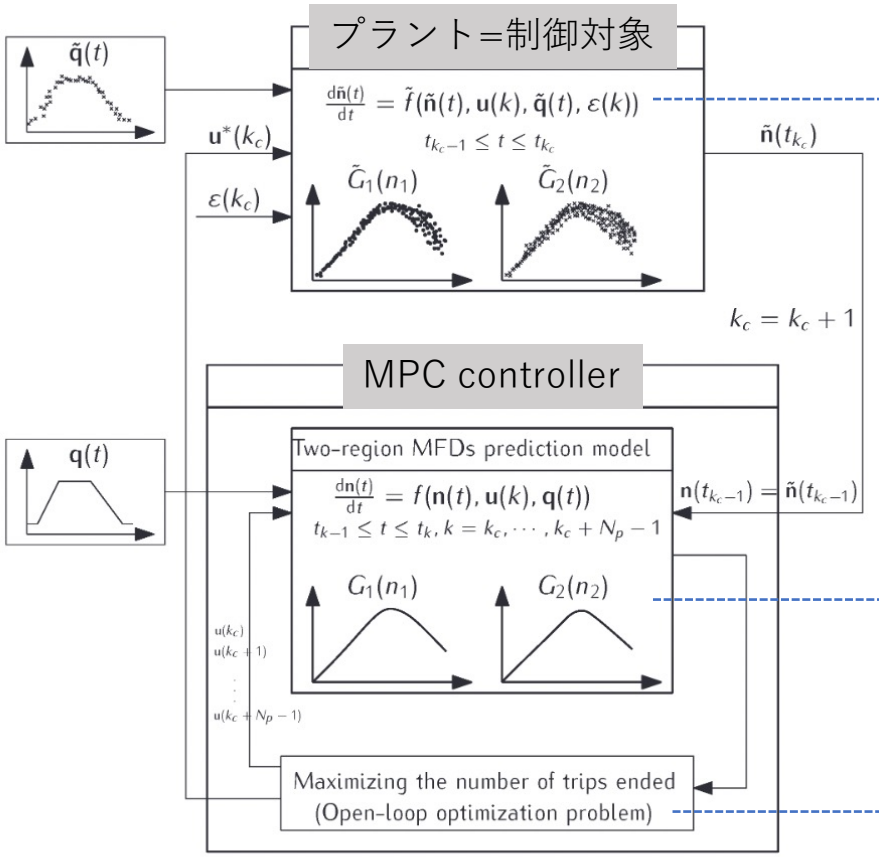
4章　MPCとGCのパフォーマンスを比較

5章　制御系列を滑らかにする2種の方法を提案

Model Predictive Control (MPC) の適用

p11の2地域MFD制御の定式化は,
- **制約付き非線形最適化問題**
- MFDの**バラつきによる誤差**に対応する必要がある

MPCを使おう！ (see p8)



制御入力 (流出入率) を加えた場合のMFDを観測

feedback構造

プラントの出力 (accumulation) を初期状態として,
将来のaccumulationを予測

予測結果をもとにトリップ完了台数を最大化するような
制御入力 (流出入率) の系列を求解

Prediction model

各時刻での初期状態と入力の系列が与えられたとき，将来の状態を予測するモデルが必要！
→ p11の微分方程式(2)～(5)を解けばいい.

（再掲）

n_{ij}の保存則

$$\frac{dn_{11}(t)}{dt} = q_{11}(t) + u_{21}(t) \cdot M_{21}(t) - M_{11}(t) \quad (2)$$

$$\frac{dn_{12}(t)}{dt} = q_{12}(t) - u_{12}(t) \cdot M_{12}(t) \quad (3)$$

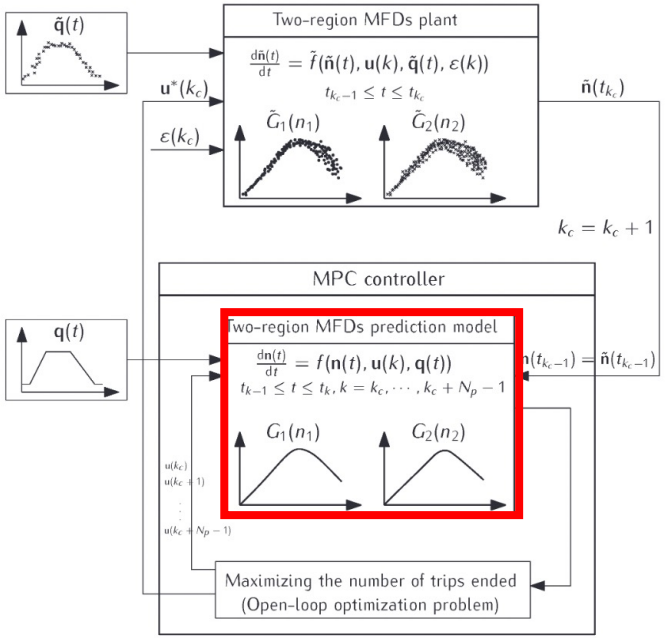
demand　流出率　1→2のフロー

$$\frac{dn_{21}(t)}{dt} = q_{21}(t) - u_{21}(t) \cdot M_{21}(t) \quad (4)$$

$$\frac{dn_{22}(t)}{dt} = q_{22}(t) + u_{12}(t) \cdot M_{12}(t) - M_{22}(t) \quad (5)$$

demand　流入率　1→2のフロー　トリップ完了した車

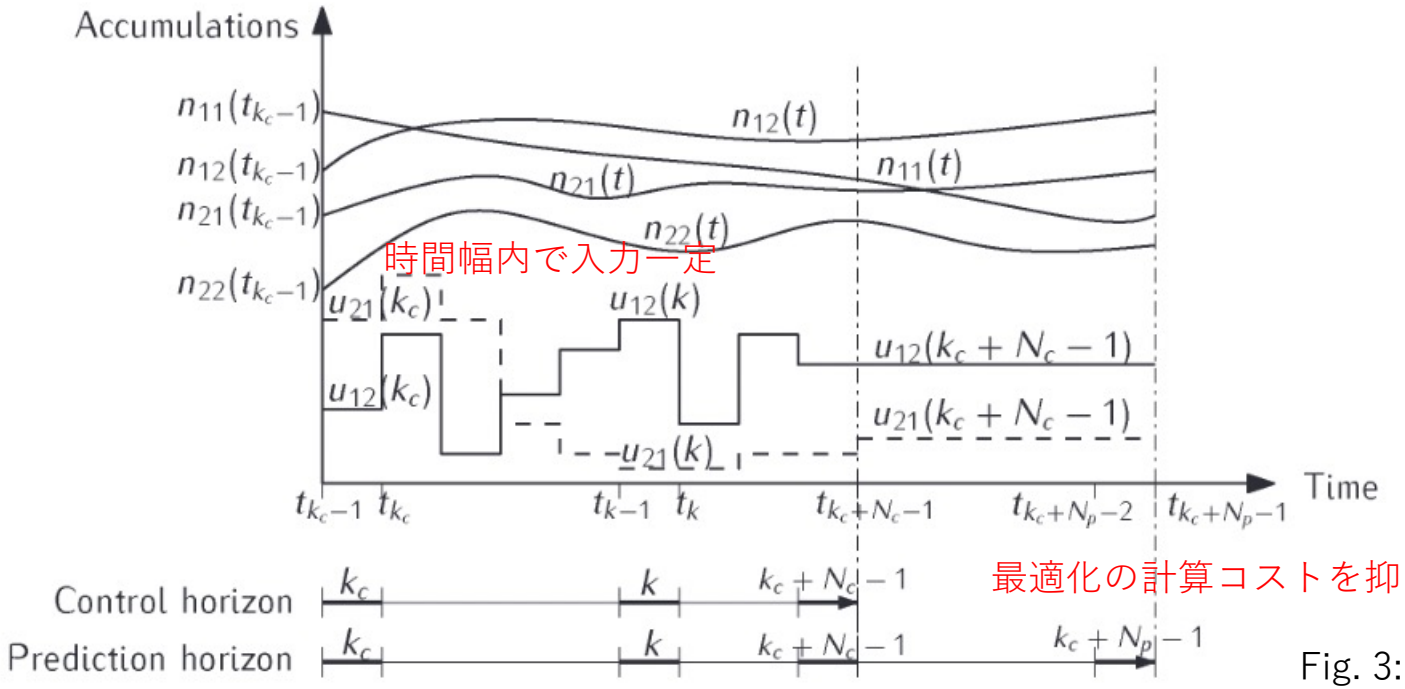
初期状態 = 前段階でのプラントの出力
将来の制御入力 = 前段階での最適化の結果
将来の交通需要 = 固定で与える

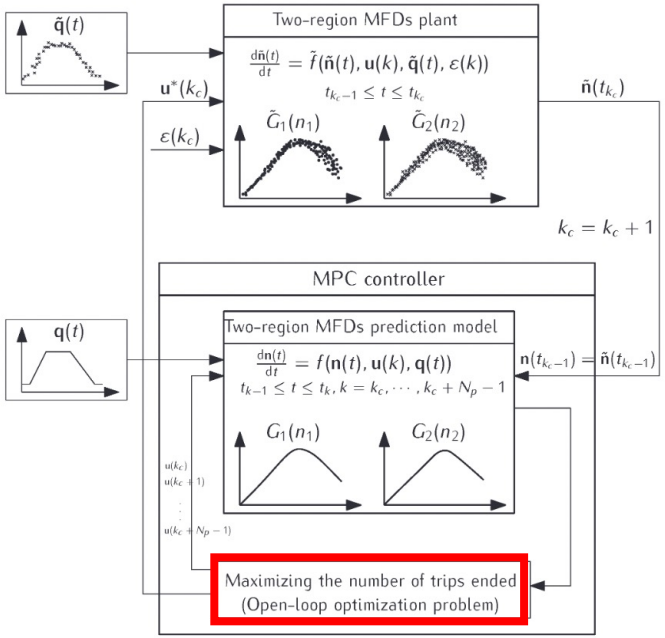


Optimization problem

直接法を使う → ”first discretize and then optimize”

- 制御の開始から終了まで、時刻を等間隔に分割
- 分割された時刻ごとに、その時間幅内で入力$u(t)$ の値は一定
- $n(t)$の常微分方程式（予測モデル）が制約条件に入るので，時間幅ごとに状態$n(t)$を数値積分により求める





最適化の計算コストを抑えるため，Control horizon < Prediction horizon

Fig. 3: direct sequential methodの説明

Optimization problem

以上の手法により，p11の2地域MFD最適制御問題は，制御入力が区間内で一定な
非線形最適化問題に近似された.

$$\min_{\mathbf{u}(k_c),\mathbf{u}(k_c+1),\ldots,\mathbf{u}(k_c+N_p-1)} \quad -z(t_{k_c+N_p-1}) \qquad (14)$$

少し形が変わったが，目的関数の内容は同じ

subject to

$$\frac{d\mathbf{n}(t)}{dt} = f(\mathbf{n}(t),\mathbf{u}(k),\mathbf{q}(t)) \quad \text{Prediction model} \qquad (15)$$

$$\frac{dz(t)}{dt} = M_{11}(t) + M_{22}(t) \qquad (16)$$

$$\mathbf{u}_{\min} \le \mathbf{u}(k) \le \mathbf{u}_{\max} \quad \text{流出入率の制御に上下限} \qquad (17)$$

where $t_{k-1} \le t \le t_k \quad k = k_c, k_c+1, \ldots, k_c+N_p-1$

$$\mathbf{u}(k) = \mathbf{u}(k_c+N_c-1) \quad k = k_c+N_c, \ldots, k_c+N_p-1 \qquad (18)$$

Control horizonより大の区間は入力一定

p11の(6)～(9)を見るとtが連続なので制約条件が無限個になるが，積分により回避.

$$\sum_{k=k_c}^{k_c+N_p-1} \int_{t_{k-1}}^{t_k} \max\{0; -n_{11}(t) - n_{12}(t)\}^2 \, dt \le \epsilon \qquad (19)$$

$$\sum_{k=k_c}^{k_c+N_p-1} \int_{t_{k-1}}^{t_k} \max\{0; -n_{21}(t) - n_{22}(t)\}^2 \, dt \le \epsilon \qquad (20)$$

$$\sum_{k=k_c}^{k_c+N_p-1} \int_{t_{k-1}}^{t_k} \max\{0; n_{11}(t)+n_{12}(t)-n_{1,\mathrm{jam}}\}^2 dt \le \epsilon \qquad (21)$$

$$\sum_{k=k_c}^{k_c+N_p-1} \int_{t_{k-1}}^{t_k} \max\{0; n_{21}(t)+n_{22}(t)-n_{2,\mathrm{jam}}\}^2 dt \le \epsilon \qquad (22)$$

プラント = 制御対象

プラントは予測モデルのMFDとは異なる.
← プラントの観測結果は**MFDのバラつき**と**交通需要のノイズ**を含むので

1. MFDのバラつき・誤差の表現

誤差項　　一様分布によって表現

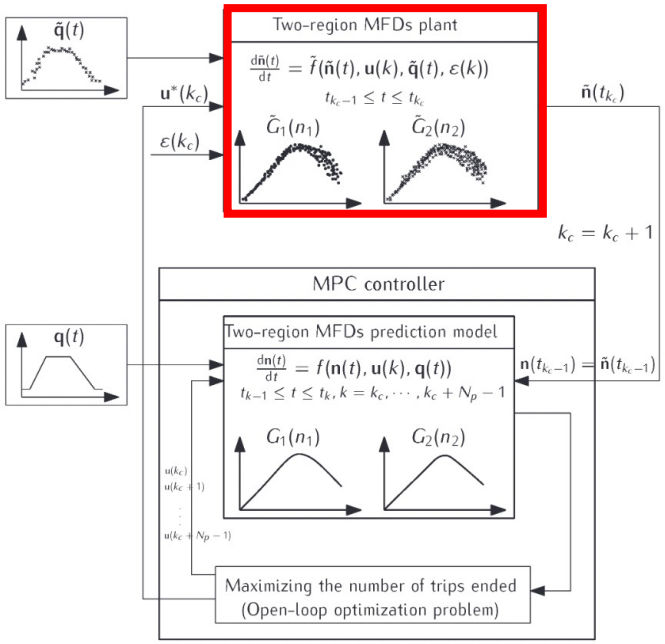
$$\varepsilon\left(\tilde{n}_1(t_{k-1})\right) \sim U\left(-\alpha_1 \cdot \tilde{n}_1(t_{k-1}),\ \alpha_1 \cdot \tilde{n}_1(t_{k-1})\right) \quad (23)$$

$$\varepsilon\left(\tilde{n}_2(t_{k-1})\right) \sim U\left(-\alpha_2 \cdot \tilde{n}_2(t_{k-1}),\ \alpha_2 \cdot \tilde{n}_2(t_{k-1})\right) \quad (24)$$

MFDのバラつきを表現

$$\tilde{G}_1\left(\tilde{n}_1(t)\right) = G_1\left(\tilde{n}_1(t)\right) + \varepsilon(\tilde{n}_1(t_{k-1})) \quad (25)$$

$$\tilde{G}_2\left(\tilde{n}_2(t)\right) = G_2\left(\tilde{n}_2(t)\right) + \varepsilon(\tilde{n}_2(t_{k-1}))\,. \quad (26)$$



プラント = 制御対象

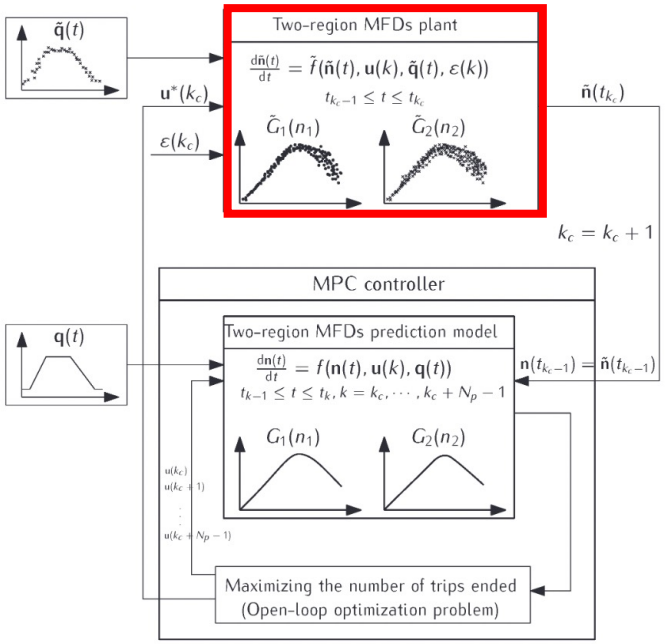
2. 交通需要のノイズ

1） Unbiased demand noise = day to dayのランダムな需要変動

正規分布によって表現

$$\tilde{q}_{ij}(t) = \max\left(q_{ij}(t) + \mathcal{N}(0, \sigma_{ij}^2), 0\right) \qquad (27)$$

2） Biased demand noise = 偶発的な需要変更（事故など）
適当に需要変動を与える



よって，プラントのaccumulationの微分方程式は以下.

$$\frac{d\tilde{\mathbf{n}}(t)}{dt} = \tilde{f}\left(\tilde{\mathbf{n}}(t), \mathbf{u}(k), \tilde{\mathbf{q}}(t), \boldsymbol{\varepsilon}(k)\right) \qquad (28)$$

where $\boldsymbol{\varepsilon}(k) = [\varepsilon(\tilde{n}_1(t_{k-1})), \varepsilon(\tilde{n}_2(t_{k-1}))]^{\mathrm{T}}$.

Greedy Controller（比較用）

Greedy Controller = 静的なフィードバック制御.
混雑状況によって，流出入を最大にするか最小にするかを選択する（原始的）.

(i) 両地域とも混雑していない場合,

$$[u_{12}(t), u_{21}(t)] = [u_{\max}, u_{\max}]$$

(ii) 地域1が混雑・地域2は混雑していない場合（逆も同様）,

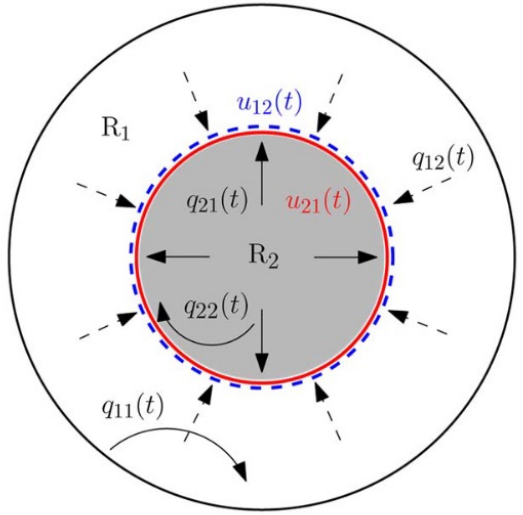
$$[u_{12}(t), u_{21}(t)] = [u_{\max}, u_{\min}]$$

(iii) 両地域とも混雑している場合,

$$[u_{12}(t), u_{21}(t)] = [u_{\max}, u_{\min}] \quad \text{if} \quad \frac{n_1(t)}{n_{1,\mathrm{jam}}} > \frac{n_2(t)}{n_{2,\mathrm{jam}}}$$

$$[u_{12}(t), u_{21}(t)] = [u_{\max}, u_{\min}] \quad \text{otherwise.}$$

混雑していないほうに多く流す

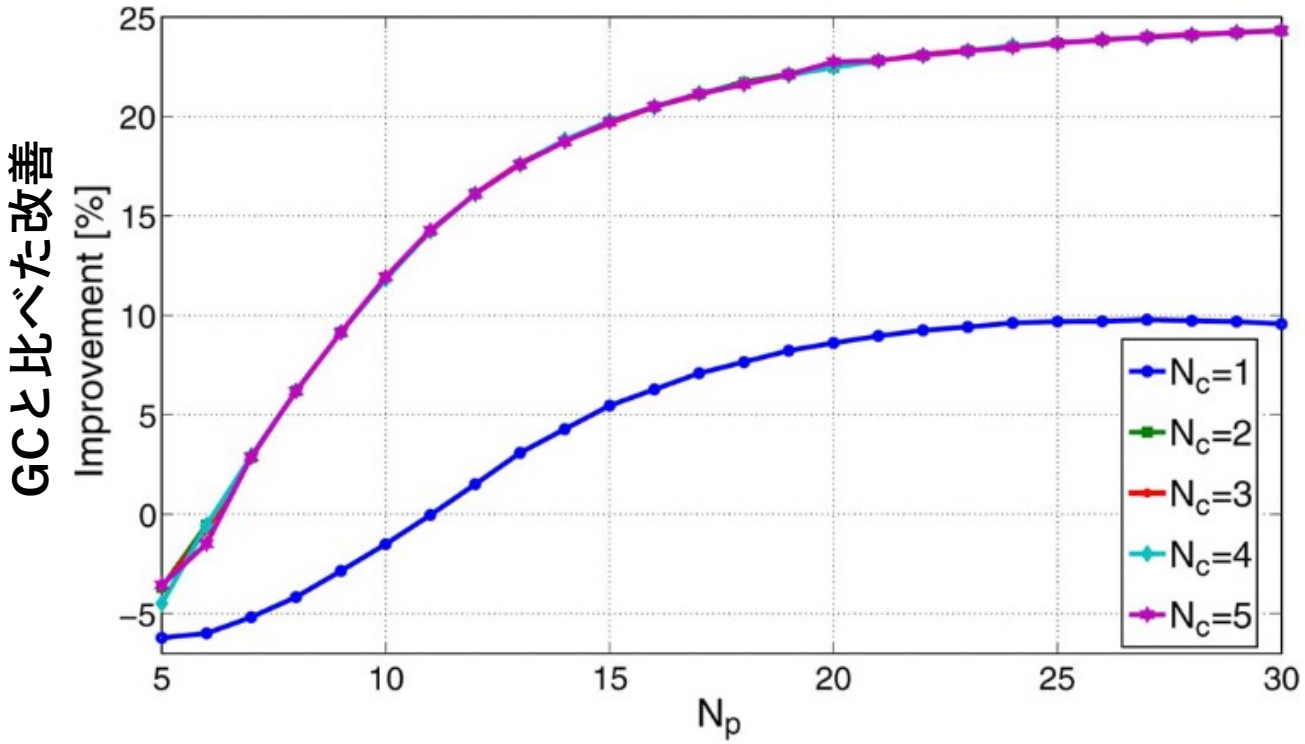


パラメータチューニング

- N_p・N_c大 → パフォーマンス向上 but 計算時間増加のトレードオフがある.
- 最適化計算が1タイムステップより長いとMPCが機能しない.

→ **バランスする予測ホライズンN_pと，制御ホライズンN_cを決める.**

→ N_p・N_cを少しずつ変化させながらMPCの結果を確認する（感度分析）



- 1タイムステップは一般的な信号間隔の60秒とした.
- 計算結果から$N_p = 20,\ N_c = 2$として構わない（それ以上増加させてもパフォーマンスが大きく向上しないから）

論文の流れ

2章　MFDによって微分方程式モデルを作り，2地域の最適流出入制御 (perimeter control) を定式化

3章　MPCを定式化，パラメータのチューニング，比較用にGreedy Controller (GC) の導入

4章　MPCとGCのパフォーマンスを比較

5章　制御系列を滑らかにする2種の方法を提案

Case Study

目的：

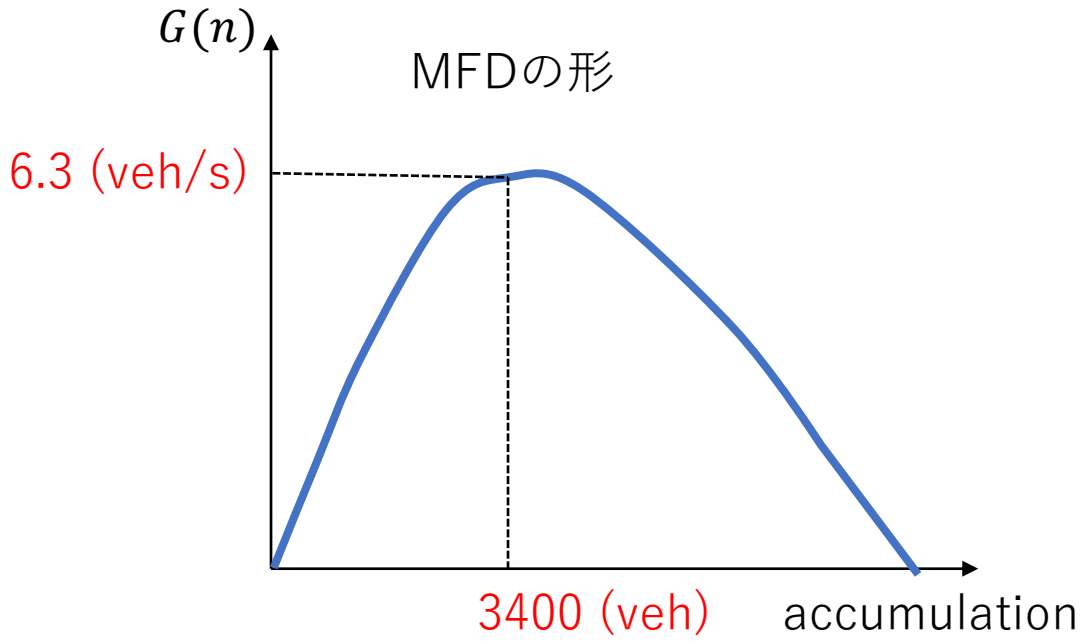
- 様々な需要レベル・MFD特性の下で，**渋滞時と非渋滞時のMPC制御の効率性**を確かめる．
- MFDのバラつきと需要変動を加えて，**MPCとGCの頑健性 (robustness)** を確認

設定：

- $N_p = 20,\ N_c = 2$
- $u_{\max} = 0.9,\ \ u_{\min} = 0.1$
- MFD関数：$G_i\left(n_i(t)\right) = a_i n_i(t)^3 + b_i n_i(t)^2 + c_i n_i(t)$

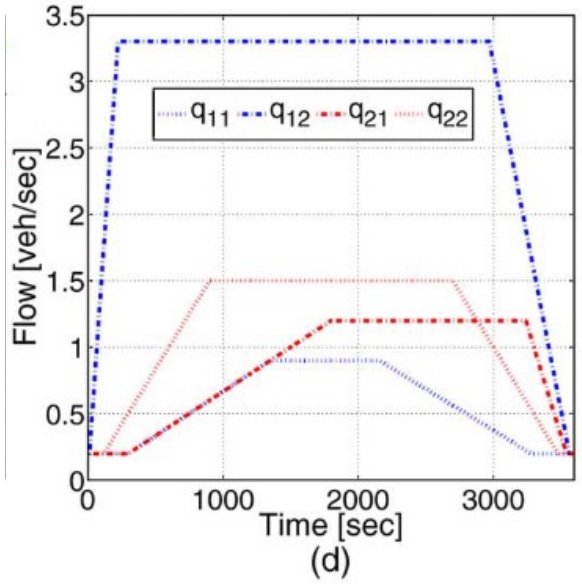
  $a_i = \frac{1.4877}{3600} \cdot 10^{-7}, b_i = -\frac{2.9815}{3600} \cdot 10^{-3}, c_i = \frac{15.0912}{3600}$

  （横浜のMFDの形）



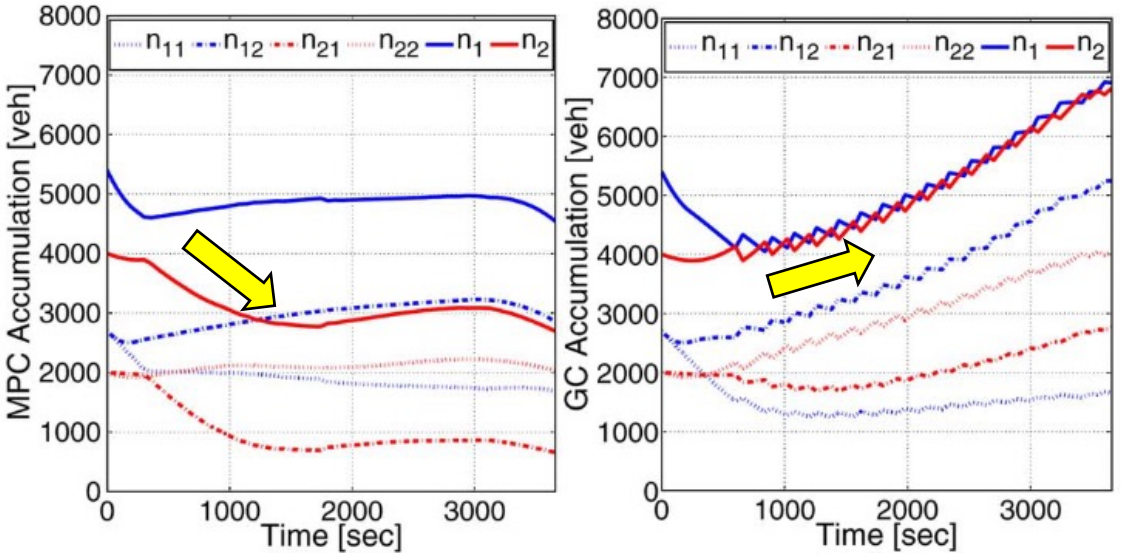
Example1: 2地域とも初期状態が混雑状態

需要変動パターン（与える）



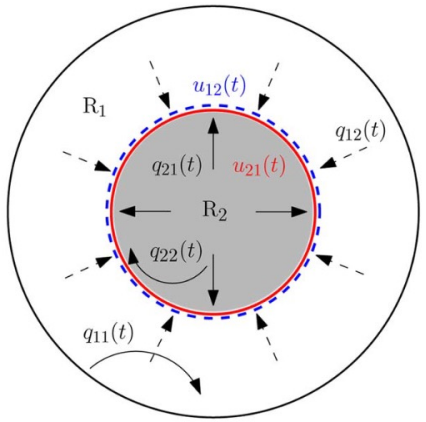
MPC・GC

2地域の車両存在台数の変化



MPC **GC**

1→2の需要が大きい
=朝ピークを再現

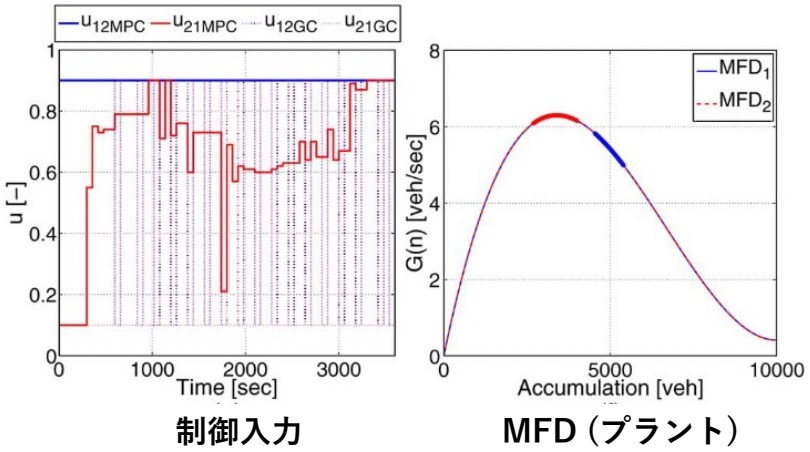


- MPCは地域2への流入を絞って車両存在台数を減少させ混雑を回避
- GCは地域1,2の流入率を等しくして混雑が悪化

←MPCは将来の需要を予測しながら制御するため

Example1: MFDのバラつきの影響

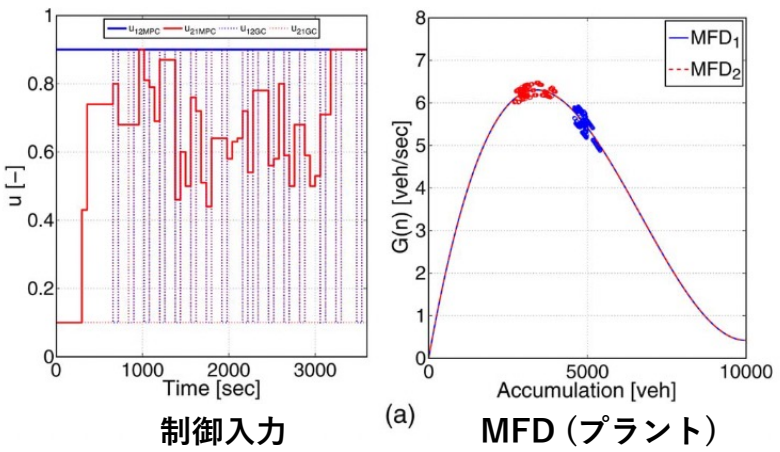
MFDのバラつき無の場合 $(\alpha_1 = \alpha_2 = 0)$



制御入力　　MFD (プラント)

トリップ完了数 (MPC) 23.55 [veh.10^3]
トリップ完了数 (GC)　17.07 [veh.10^3]

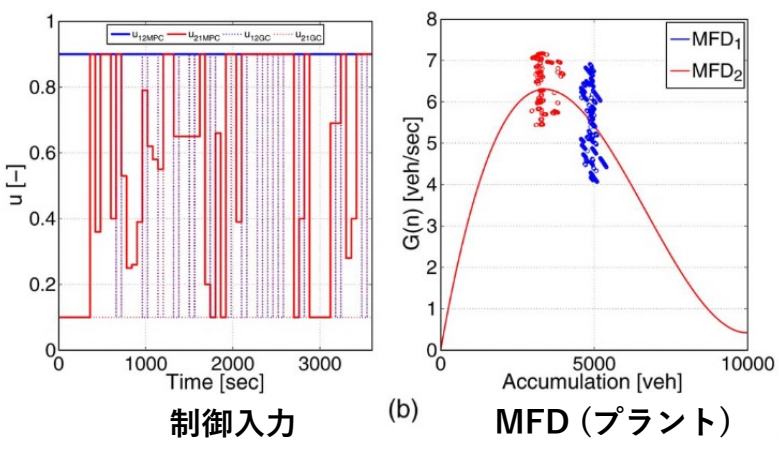
MFDのバラつき小の場合 $(\alpha_1 = \alpha_2 = 0.2)$



制御入力　(a)　MFD (プラント)

トリップ完了数 (MPC) 23.63 [veh.10^3]
トリップ完了数 (GC)　17.11 [veh.10^3]

MFDのバラつき大の場合 $(\alpha_1 = \alpha_2 = 1.0)$



制御入力　(b)　MFD (プラント)

トリップ完了数 (MPC) 24.04 [veh.10^3]
トリップ完了数 (GC)　17.21 [veh.10^3]

- プラントのMFDのバラつきが多くなるほど，$u_{21}(t)$ (図中赤線)が**滑らかでなくなる**
  →×滑らかでない制御は実適用が困難・境界付近のNWの均質性を崩す・MFDのバラつきを大きくする
- MPCのパフォーマンスはそれほど低下しない
  → **MFDが粗い近似でも効果が期待できる**

Example2,3: 需要レベルの影響

Example1

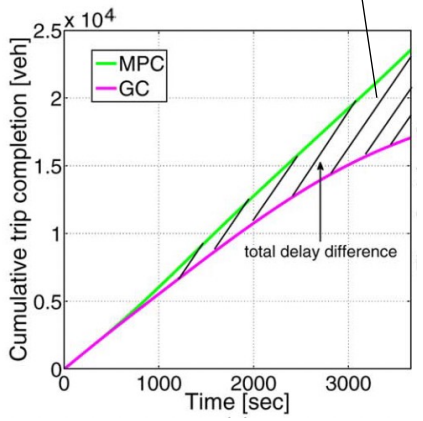
Example2
=各需要を16%カット

Example3
=各需要を32%カット

Example 1	without errors ($\alpha_1 = \alpha_2 = 0$)			small errors ($\alpha_1 = \alpha_2 = 0.2$)			large errors ($\alpha_1 = \alpha_2 = 1$)		
	MPC ($veh.10^3$)	GC ($veh.10^3$)	MPC-GC ($veh.s.10^3$)	MPC	GC	MPC-GC	MPC	GC	MPC-GC
without noises	23.55	17.07	7791.6 (22.5%)	23.63	17.11	7883.5 (22.7%)	24.04	17.21	8370.0 (24.1%)
low noises	23.37	16.78	7864.3 (22.9%)	23.48	16.82	7997.3 (23.3%)	23.93	17.00	8379.5 (24.3%)
high noises	22.80	16.07	7892.6 (23.5%)	23.02	16.35	7931.9 (23.4%)	23.34	15.84	9146.5 (27.6%)
Example 2	without errors ($\alpha_1 = \alpha_2 = 0$)			small errors ($\alpha_1 = \alpha_2 = 0.2$)			large errors ($\alpha_1 = \alpha_2 = 1$)		
	MPC	GC	MPC-GC	MPC	GC	MPC-GC	MPC	GC	MPC-GC
without noises	24.11	20.49	6536.8 (17.3%)	24.12	20.51	6548.8 (17.3%)	24.13	20.56	6664.4 (17.6%)
low noises	24.15	20.43	6654.2 (17.7%)	24.15	20.47	6666.2 (17.7%)	24.17	20.56	6741.1 (17.9%)
high noises	24.29	20.28	6885.2 (18.4%)	24.30	20.34	6899.7 (18.4%)	24.33	20.45	6966.9 (18.6%)
Example 3	without errors ($\alpha_1 = \alpha_2 = 0$)			small errors ($\alpha_1 = \alpha_2 = 0.2$)			large errors ($\alpha_1 = \alpha_2 = 1$)		
	MPC	GC	MPC-GC	MPC	GC	MPC-GC	MPC	GC	MPC-GC
without noises	21.70	21.63	1789.3 (4.4%)	21.70	21.64	1736.1 (4.2%)	21.70	21.64	1836.7 (4.5%)
low noises	21.78	21.69	1923.3 (4.7%)	21.78	21.69	1985.8 (4.9%)	21.78	21.70	2051.6 (5.0%)
high noises	22.11	21.94	2466.7 (6.1%)	22.11	21.94	2490.4 (6.1%)	22.11	21.95	2578.9 (6.4%)

総遅れ時間 (total delay) = MPCの代わりにGCを使うことによる総遅れ時間
を使って評価する.

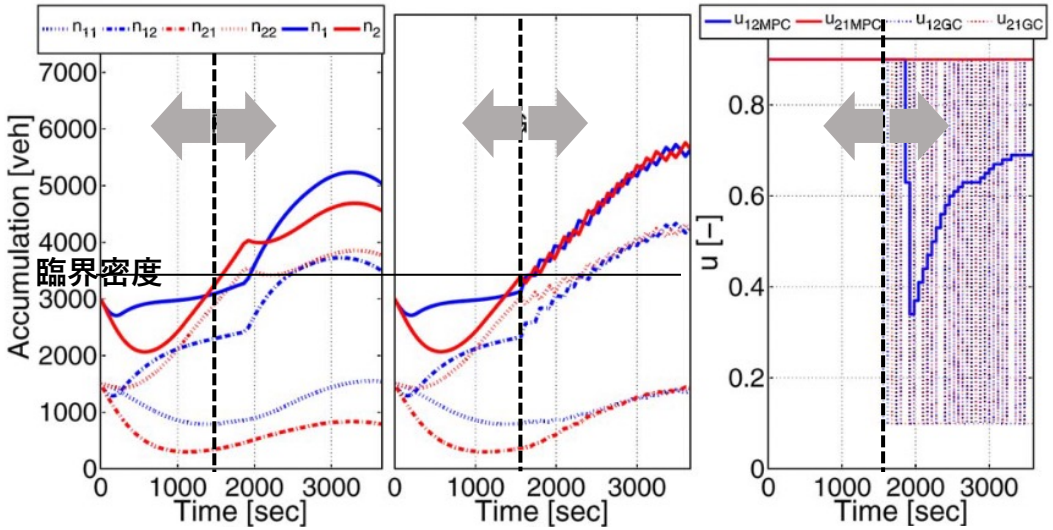
- 総遅れ時間が全て正より，**MPCのGCに対する優位性**がわかる.
- 混雑している状況ほどMPCとGCの差は大きい. 青枠より，**混雑していない状況ではMPCとGCのパフォーマンスはほぼ同じ**



Example4: 需要レベルの影響

Example4
=初め非混雑から混雑へ

Example 4	without errors ($\alpha_1 = \alpha_2 = 0$)			small errors ($\alpha_1 = \alpha_2 = 0.2$)			large errors ($\alpha_1 = \alpha_2 = 1$)		
	MPC	GC	MPC-GC	MPC	GC	MPC-GC	MPC	GC	MPC-GC
without noises	24.40	22.72	1143.7 (2.6%)	24.40	22.78	1135.4 (2.6%)	24.43	22.96	1120.4 (2.5%)
low noises	24.35	22.61	1200.0 (2.7%)	24.36	22.66	1204.5 (2.7%)	24.39	22.85	1192.8 (2.7%)
high noises	24.25	22.26	1439.3 (3.3%)	24.29	22.31	1435.9 (3.3%)	24.28	21.85	2032.9 (4.7%)

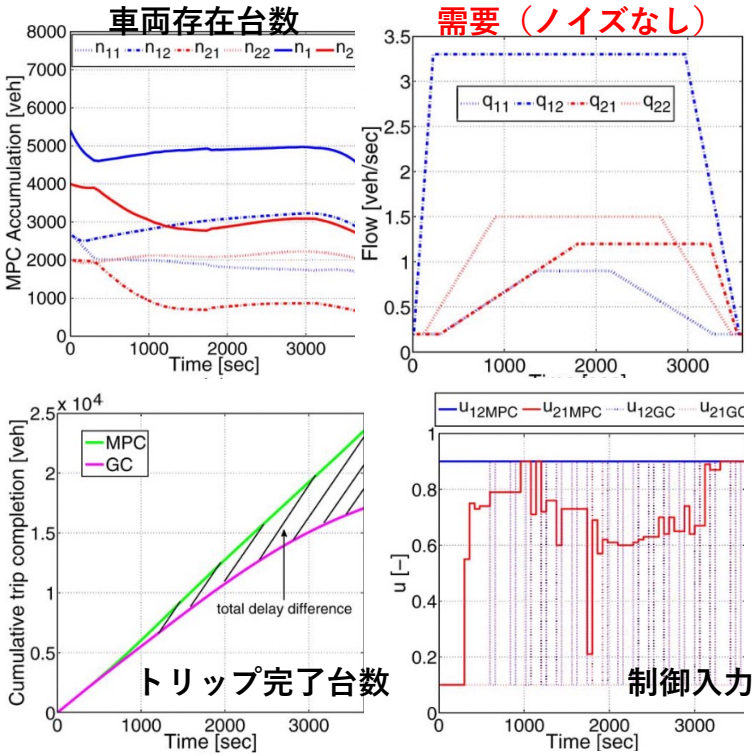


車両存在台数 (MPC)　車両存在台数 (GC)　制御入力

- 点線の前 (非混雑) はMPCとGCの挙動は似ているが，点線の後 (混雑)では全く違う
  → **混雑状態ではMPCとGCの制御が大きく異なる**

Example1: 需要の unbiased ノイズの影響

unbiased ノイズ無 ($\sigma_{ij} = 0$)

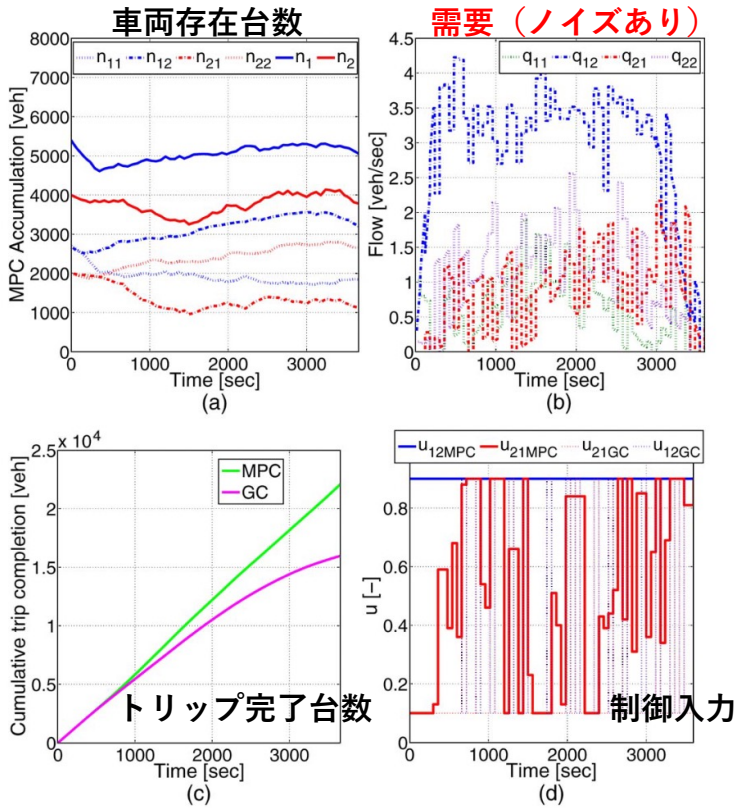


トリップ完了台数
23.55 [veh.10³]

トリップ完了台数

制御入力

unbiased ノイズ有 ($\sigma_{ij} = 0.5$)



トリップ完了台数
22.80 [veh.10³]

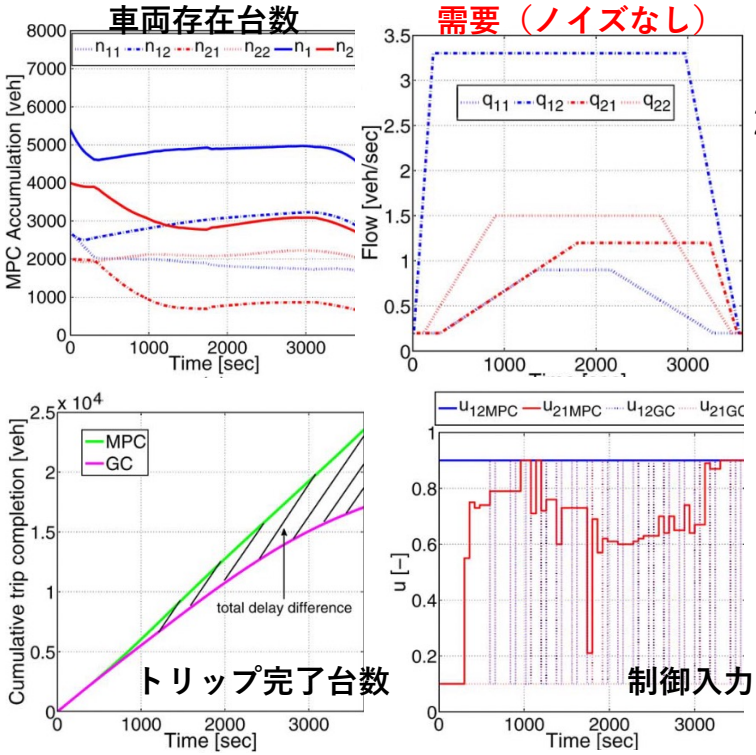
トリップ完了台数

制御入力

- **MPCのパフォーマンスはunbiased ノイズの影響をあまり受けない**
- **需要のunbiased ノイズがあると制御入力の乱れが大きくなる**

Example1: 需要の biased ノイズの影響

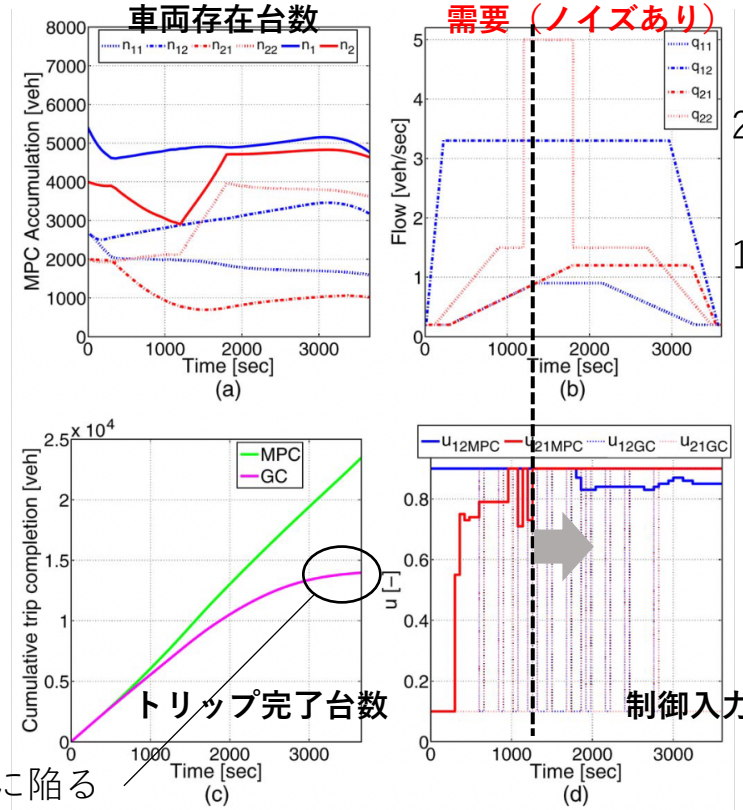
biased ノイズ無



トリップ完了数 (MPC)
23.55 [veh.10^3]

トリップ完了数 (GC)
17.07 [veh.10^3]

biased ノイズ有 (急激な需要変化)



トリップ完了数 (MPC)
23.29 [veh.10^3]

トリップ完了数 (GC)
13.97 [veh.10^3]

需要に急なノイズ（地域2内需要の急増）が発生した後，2への流入を減らし，2からの流出を最大にして対応.

GCはgridlockに陥る

- **MPCのパフォーマンスはbiased ノイズの影響をあまり受けない**

論文の流れ

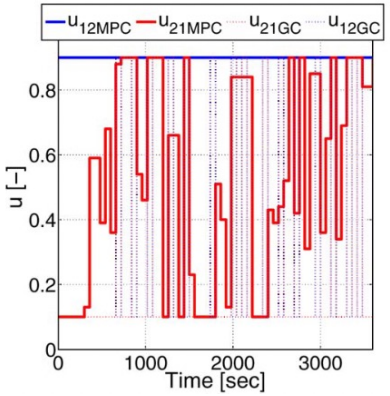
2章　MFDによって微分方程式モデルを作り，2地域の最適流出入制御 (perimeter control) を定式化

3章　MPCを定式化，パラメータのチューニング，比較用にGreedy Controller (GC) の導入

4章　MPCとGCのパフォーマンスを比較

5章　制御系列を滑らかにする2種の方法を提案

滑らかな制御を目指す



左図のように連続する2つの制御入力が大きく異なると，

- 現実への適用が困難・危険
- NWの異質性が大きくなりMFDのバラつきが大きくなる

→ 制御入力を滑らかにしたい！

→A. 入力に制約を設ける　or　B. 目的関数を修正する

A. 入力に制約を設ける

p16の最適化の式に以下の制約条件を追加

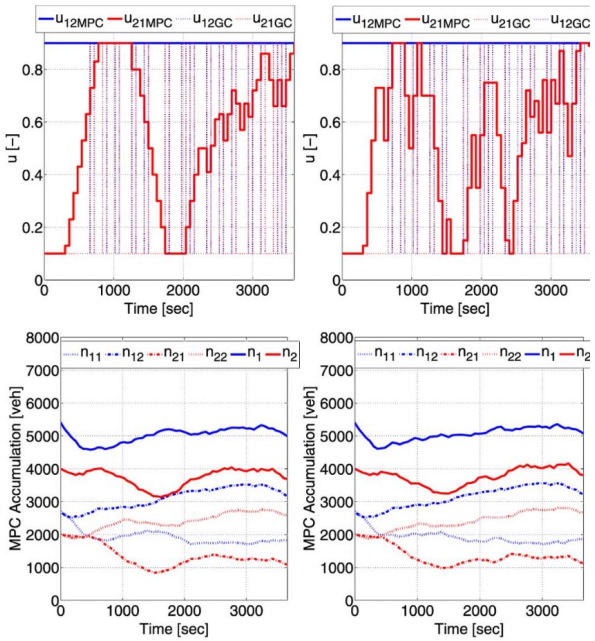
$$|u_{12}(k) - u_{12}(k-1)| \leq u_{\text{jump}} \qquad (29)$$

$$|u_{21}(k) - u_{21}(k-1)| \leq u_{\text{jump}} \qquad (30)$$

連続ステップの制御入力の差に制限を設ける

解く

$u_{jamp} = 0.1$　$u_{jamp} = 0.2$



- 滑らかになった！
- u_{jamp}の値に対して頑健

滑らかな制御を目指す

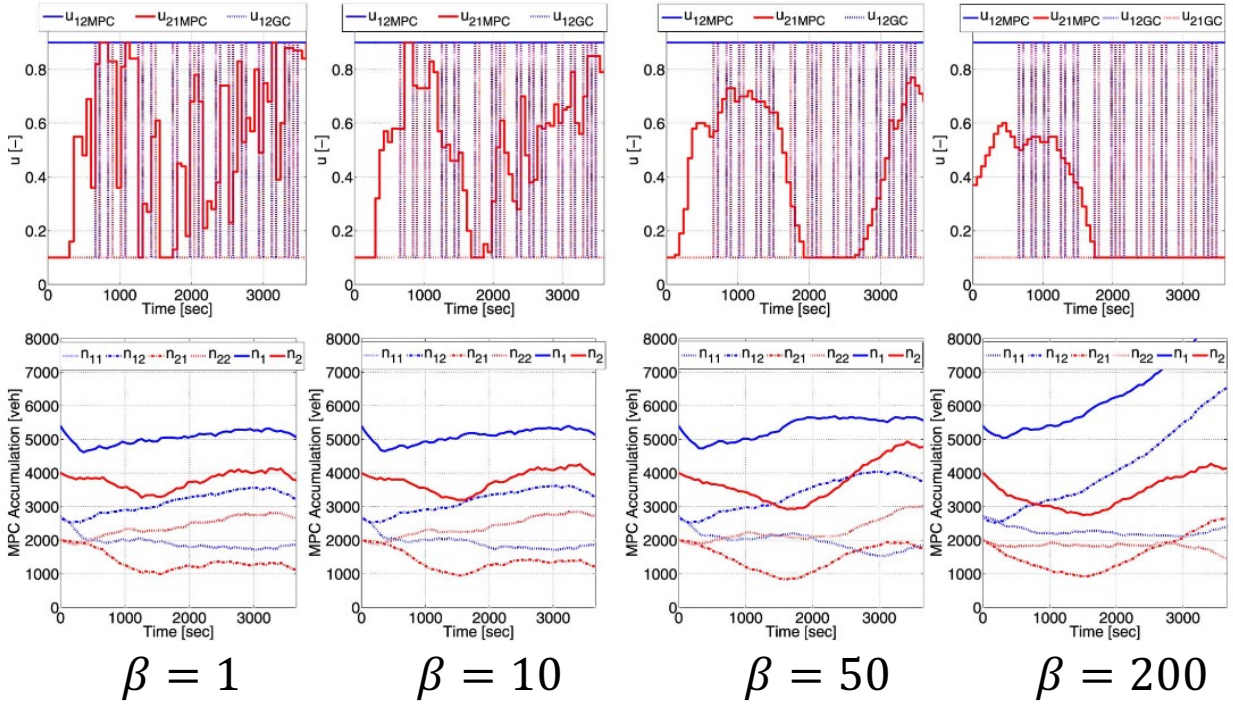
B. 目的関数を修正する

p16の最適化の目的関数にトレードオフを導入

$$\min_{\mathbf{u}(k_c),\ldots,\mathbf{u}(k_c+N_p-1)} \left\{ -z(t_{k_c+N_p-1}) + \beta \sum_{k=k_c}^{k_c+N_c-1} \left(|u_{12}(k)-u_{12}(k-1)|^2 + |u_{21}(k)-u_{21}(k-1)|^2 \right) \right\}$$

重み

連続ステップの制御入力の差
→最小化ゆえこの項を大きくしすぎないように計算する



$\beta = 1$　$\beta = 10$　$\beta = 50$　$\beta = 200$

- β大ほど滑らかさを重視するので，制御入力が滑らかになる
- 滑らかさとトリップ完了数最大化のトレードオフを考えると$\beta = 10$が一番いい．

まとめ

成果

✓ **MFDを用いた2地域の流出入制御 (perimeter control)**を定式化し，**MPC (Model predictive control)**で解いた.

✓ 全ての例において，**MPCはGC (Greedy Control)に勝る性能**を示した.

✓ 渋滞状態での信号制御についてシンプルでロバストな手法を示した重要な結果だ.

✓ 異質性のNWでも，均一なサブNWに分割して本手法を適用できる.

課題

△ 2地域以上では状態変数が増えるだけでなく経路選択モデルも組み込む必要
→MPCを現実的な時間内で解けるかが課題になる

△ 境界付近で大量の車を待機させると，その地域の密度の異質性によりMFDに影響する
→境界付近でうまく車を流すために，microsimulationを用いた地域内の信号制御も重要な研究課題

△ MFDは高速道路の交通の表現としては適さないので，一般道と高速の入り混じったNWではメソモデルに組み込めるのではないか

参考文献

Geroliminis, N., & Daganzo, C. F. (2008). Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings. *Transportation Research Part B: Methodological*, *42*(9), 759-770.

Daganzo, C. F. (2007). Urban gridlock: Macroscopic modeling and mitigation approaches. Transportation Research Part B: Methodological, 41(1), 49-62.

モデル予測制御（MPC）Part1 ～ 基本的な考え方 https://jp.mathworks.com/videos/introduction-to-model-predictive-control-mpc-part-1-1494946684184.html