1 電力系統モデルの数値シミュレーション

本章では，非線形の微分代数方程式系で記述される電力系統モデルの数値シミュレーション手法を解説する。また，構造化された数値シミュレーション環境を構築するための指針を示す。本章の構成は以下の通りである。まず，**??**節では，電力系統モデルの時間応答を計算することの難しさを説明する。つぎに，**??**節では，電力系統モデルの平衡状態を数値的に探索するプロセスである潮流計算について説明する。また，潮流計算により定められた母線の電圧や電力の定常値に整合するように発電機の内部状態の定常値や負荷モデルの定数を定める方法を**??**節で議論する。**??**節では，初期値や負荷の変動，地絡に対する時間応答の計算方法を説明すると同時に，実際に時間応答の計算を例示する。さいごに，**??**節では発展的な話題として，定常潮流状態における母線電圧の同期現象を数学的に解析する。

1.1 電力系統モデルの時間応答を計算するためには

1.1.1 時間応答を計算することの難しさ

??章では，発電機モデルや負荷モデルが送電網モデルで結合されることにより，電力系統全体の数理モデルが非線形の微分代数方程式系で記述されることを説明した。したがって，電力系統モデルの時間応答は，適当に設定された初期値や外部入力のもとで，その微分代数方程式を数値積分すれば求められる。しかしながら，電力系統モデルの数値シミュレーションでは

- 外部入力を正しく設定しない限り需要と供給が均衡しないため，角周波

数偏差の定常値が 0 にならず，発電機の回転子偏角が変化し続ける

- 角周波数偏差の定常値が 0 となるような外部入力値の組み合わせは無数に存在するため，現実的に妥当な外部入力値を指定しなければならない
- 母線群の電圧フェーザや電流フェーザの値は，発電機群の状態変数に対する従属変数として整合するように決定しなければならない

など，電力系統に特有の性質を考慮する必要がある。このため，MATLAB に標準実装されている微分代数方程式ソルバーを単純に使用するだけでは，電力系統の数値シミュレーションを正しく実行することはできない。この点が電力系統モデルの時間応答の計算を難しくする一因となる。

1.1.2 計 算 手 順

非線形微分代数方程式で記述される電力系統モデルの時間応答の標準的な計算手順は，つぎの 3 ステップに分けられる。

(A) 解析する需要と供給が均衡した電力系統状態を指定するため，送電網から定まるアドミタンス行列を用いて，定常状態におけるすべての母線の電流フェーザと電圧フェーザの値を計算する。

(B) 決定された母線の電流フェーザと電圧フェーザの定常値に整合するように，各発電機の内部電圧や回転子偏角の定常値，発電機への外部入力値，各負荷のインピーダンス値などを逆算する。

(C) ステップ A とステップ B で計算された需給が均衡した電力系統状態を初期値として，発電機の内部状態に摂動を与える，母線の電圧を地絡させる，負荷のパラメータ値を変化させる，などの大小様々な外乱を生じさせた場合の時間応答を計算する。

システム制御工学の観点では，ステップ A は「無数に存在する平衡点の中から数値的な解析を行う平衡点を 1 つ定めること」と理解することができる。**??** 節で後述するように，すべての母線における電流フェーザと電圧フェーザの定常値が与えられたときに，それらを実現する発電機の内部状態の定常値や外部入力値，負荷のパラメータ値は必ず存在する。このことから，すべての母線に

おける電流フェーザと電圧フェーザの定常値を計算することは，微分代数方程式で表される電力系統モデルの平衡点を計算することに等しい。電力系統工学では，このプロセスのことを**潮流計算**（power flow calculation）と呼ぶ。なお，各母線の電流フェーザと電圧フェーザの定常値を求めることは，各機器から母線に供給される有効電力と無効電力の定常値を求めることと数学的に等価である。

ステップ B では，母線の電流フェーザや電圧フェーザの定常値に整合するように発電機の内部状態の定常値や外部入力値，負荷のパラメータ値を逆算する。このステップで注意すべき点は，負荷のパラメータ値，すなわち，負荷の数理モデルを「ステップ A の潮流計算の結果から逆算して定める」という間接的な手順を踏むことである。例えば，ある母線に接続する負荷を定インピーダンスモデルに設定したい場合には，潮流計算で求められた母線の電流フェーザを電圧フェーザで除した値として，負荷のインピーダンス値を逆算するという手順を踏む。負荷モデルのパラメータを所望の値に設定したい場合には，ステップ C において，負荷のパラメータ変動に対する時間応答を計算しながら，負荷のパラメータ値を所望の値に変化させるなどの方法を用いる。

さいごに，ステップ C では，目的に応じた様々な条件で電力系統モデルの時間応答を計算する。例えば，ステップ A と B で計算された各発電機への外部入力値や負荷のパラメータ値を定数としてモデルに設定し，発電機の内部状態に適当な初期値を与えて時間応答を計算すれば，時間の経過にしたがって，発電機の内部状態はステップ B で計算された定常状態に漸近的に収束する。ただし，漸近収束が成り立つような妥当な初期値を設定するためには，ステップ A と B のように，解析の基準となる平衡点をあらかじめ計算しておく必要がある。また，計算された平衡点は適切な意味で安定でなければならない。

1.2 定常状態を数値的に探索する潮流計算

本節では，**??**節のステップ A として説明された，電力系統の定常状態を数

値的に探索する潮流計算の概要や MATLAB による実装方法を述べる。以下では，与えられたアドミタンス行列 $\boldsymbol{Y}$ に対して

$$\begin{bmatrix} \boldsymbol{I}_1(t) \\ \vdots \\ \boldsymbol{I}_N(t) \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{Y}_{11} & \cdots & \boldsymbol{Y}_{1N} \\ \vdots & \ddots & \vdots \\ \boldsymbol{Y}_{N1} & \cdots & \boldsymbol{Y}_{NN} \end{bmatrix}}_{\boldsymbol{Y}} \begin{bmatrix} \boldsymbol{V}_1(t) \\ \vdots \\ \boldsymbol{V}_N(t) \end{bmatrix} \tag{1.1}$$

を満たす母線群の電流フェーザと電圧フェーザの分布

$$\big(|\boldsymbol{I}_1(t)|, \angle\boldsymbol{I}_1(t), |\boldsymbol{V}_1(t)|, \angle\boldsymbol{V}_1(t), \ldots |\boldsymbol{I}_N(t)|, \angle\boldsymbol{I}_N(t), |\boldsymbol{V}_N(t)|, \angle\boldsymbol{V}_N(t)\big) \tag{1.2}$$

を時刻 t での**潮流状態**（power flow distribution）と呼ぶ。なお，各々の電流フェーザと電圧フェーザは時変であり，電流と電圧の物理法則から，任意の時刻 t においてそれらは式**??**の方程式を満たさなければならない。

潮流計算は，「定常的な潮流状態の 1 つ」を求める計算プロセスである。ここで，定常的な潮流状態とは，ある定数の電流フェーザ $\boldsymbol{I}_i^\star$ と電圧フェーザ $\boldsymbol{V}_i^\star$ に対して，すべての母線 i について

$$\boldsymbol{I}_i(t) = \boldsymbol{I}_i^\star, \qquad \boldsymbol{V}_i(t) = \boldsymbol{V}_i^\star, \qquad \forall t \geqq 0$$

が成り立つことを指す。また，母線に供給される有効電力と無効電力の定義

$$P_i(t) + \boldsymbol{j}Q_i(t) = \boldsymbol{V}_i(t)\overline{\boldsymbol{I}}_i(t) \tag{1.3}$$

を用いて電流フェーザを消去することにより，式**??**の連立方程式が

$$\begin{cases} P_1(t) + \boldsymbol{j}Q_1(t) = \displaystyle\sum_{j=1}^{N} \overline{\boldsymbol{Y}}_{1j}|\boldsymbol{V}_1(t)||\boldsymbol{V}_j(t)|e^{\boldsymbol{j}\left(\angle\boldsymbol{V}_1(t) - \angle\boldsymbol{V}_j(t)\right)} \\ \qquad\qquad \vdots \\ P_N(t) + \boldsymbol{j}Q_N(t) = \displaystyle\sum_{j=1}^{N} \overline{\boldsymbol{Y}}_{Nj}|\boldsymbol{V}_N(t)||\boldsymbol{V}_j(t)|e^{\boldsymbol{j}\left(\angle\boldsymbol{V}_N(t) - \angle\boldsymbol{V}_j(t)\right)} \end{cases} \tag{1.4}$$

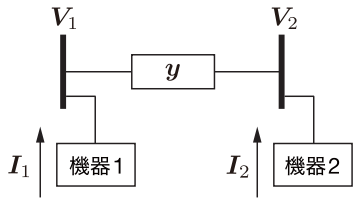


図 1.1 2 つの母線で構成される電力系統モデル

とも等価であることがわかる。したがって，文脈に応じて，式**??**を満たす有効電力，無効電力，電圧フェーザの分布

$$\big(P_1(t), Q_1(t), |\boldsymbol{V}_1(t)|, \angle \boldsymbol{V}_1(t), \ldots, P_N(t), Q_N(t), |\boldsymbol{V}_N(t)|, \angle \boldsymbol{V}_N(t)\big) \tag{1.5}$$

も同じく時刻 t での潮流状態と呼ぶ。

1.2.1 潮流計算の概要

2 つの母線で構成される簡単な例を用いて潮流計算の特徴を説明しよう。

例 1.1 (2 つの母線で構成される電力系統モデルの潮流計算) **??**の 2 つの母線から構成される電力系統を考えよう。各母線には負荷または発電機が接続されているものと考えるが，潮流計算では接続される機器の種類を具体的に指定する必要はない。

2 つの母線を結ぶ送電線のアドミタンスを $\boldsymbol{y} \in \mathbb{C}$ とする。基本的な送電線モデルを採用するとき，母線の電圧フェーザと電流フェーザには

$$\begin{bmatrix} \boldsymbol{I}_1^\star \\ \boldsymbol{I}_2^\star \end{bmatrix} = \begin{bmatrix} \boldsymbol{y} & -\boldsymbol{y} \\ -\boldsymbol{y} & \boldsymbol{y} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_1^\star \\ \boldsymbol{V}_2^\star \end{bmatrix} \tag{1.6}$$

の関係が成り立つ。ただし，定常状態における値であることを明示するために「$\star$」をつけて表記した。ここで，式**??**の関係を用いて電流フェーザを消去すれば，式**??**と等価な連立方程式が，定常状態における有効電力，無効電力，電圧フェーザを用いて

$$\left\{\begin{array}{l} P_1^\star + \boldsymbol{j} Q_1^\star = \overline{\boldsymbol{y}} \left(|\boldsymbol{V}_1^\star|^2 - |\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| e^{\boldsymbol{j}(\angle \boldsymbol{V}_1^\star - \angle \boldsymbol{V}_2^\star)} \right) \\ P_2^\star + \boldsymbol{j} Q_2^\star = \overline{\boldsymbol{y}} \left(|\boldsymbol{V}_2^\star|^2 - |\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| e^{\boldsymbol{j}(\angle \boldsymbol{V}_2^\star - \angle \boldsymbol{V}_1^\star)} \right) \end{array}\right. \tag{1.7a}$$

のように得られる。潮流計算の目的は，この連立方程式を満たす 1 組の

$$(P_1^\star, Q_1^\star, |\boldsymbol{V}_1^\star|, \angle \boldsymbol{V}_1^\star, P_2^\star, Q_2^\star, |\boldsymbol{V}_2^\star|, \angle \boldsymbol{V}_2^\star)$$

を定めることである。

送電線のコンダクタンスとサセプタンスを

$$g := \mathsf{Re}[\boldsymbol{y}], \qquad b := \mathsf{Im}[\boldsymbol{y}]$$

と表す。このとき，式**??**の実部と虚部に関する方程式を考えると

$$\left\{\begin{array}{l} P_1^\star = g|\boldsymbol{V}_1^\star|^2 - g|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \cos \angle \boldsymbol{V}_{12}^\star - b|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \sin \angle \boldsymbol{V}_{12}^\star \\ P_2^\star = g|\boldsymbol{V}_2^\star|^2 - g|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \cos \angle \boldsymbol{V}_{21}^\star - b|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \sin \angle \boldsymbol{V}_{21}^\star \\ Q_1^\star = -b|\boldsymbol{V}_1^\star|^2 + b|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \cos \angle \boldsymbol{V}_{12}^\star - g|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \sin \angle \boldsymbol{V}_{12}^\star \\ Q_2 = -b|\boldsymbol{V}_2^\star|^2 + b|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \cos \angle \boldsymbol{V}_{21}^\star - g|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \sin \angle \boldsymbol{V}_{21}^\star \end{array}\right. \tag{1.7b}$$

という 4 本の連立方程式が得られる。ただし，$\angle \boldsymbol{V}_{ij}^\star$ は $\angle \boldsymbol{V}_i^\star - \angle \boldsymbol{V}_j^\star$ を表す。電圧フェーザの位相は差分値のみが意味をもつため，実質的に決定すべき変数は 7 個である。したがって，式**??**の方程式には 3 変数分の自由度が存在する。これらを決定するためには，例えば，$(|\boldsymbol{V}_1^\star|, |\boldsymbol{V}_2^\star|, \angle \boldsymbol{V}_{12}^\star)$ の 3 変数を適当な値に指定すれば良い。これにより，残りの $(P_1^\star, P_2^\star, Q_1^\star, Q_2^\star)$ は式**??**の右辺を計算するだけで決定される。しかしながら，この方法では，各母線の電圧フェーザは任意の値に設定できる一方で，各母線に供給される電力や消費される電力を任意の値に設定することができない。すなわち，どれが電力を消費する負荷母線であり，どれが電力を供給する発電機母線であるかを指定することができない。したがって，現実的な設定で数値シミュレーションを実行するためには，指定した有効電力や無効電力の値を実現するように電圧フェーザの値を適切に定めることがしばしば必要となる。

例えば，母線に供給される有効電力がバランスした

$$P_1^\star = 1, \qquad P_2^\star = -1 \tag{1.8}$$

を実現する $(|\boldsymbol{V}_1^\star|, |\boldsymbol{V}_2^\star|, \angle\boldsymbol{V}_{12}^\star)$ を求めることを考えてみよう。これは，母線 1 に接続された機器が供給する有効電力と母線 2 に接続された機器が消費する有効電力が，定常状態においてどちらも 1 である場合に，電圧フェーザの分布を求めることに相当する。式**??**の $P_1^\star$ と $P_2^\star$ に関する方程式を足し上げれば，$P_1^\star$ と $P_2^\star$ の和は 0 であることから

$$\begin{aligned} 0 &= g\Big\{|\boldsymbol{V}_1^\star|^2 + |\boldsymbol{V}_2^\star|^2 - 2|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star| \cos\angle\boldsymbol{V}_{12}^\star\Big\} \\ &= g\Big\{\big(|\boldsymbol{V}_1^\star| - |\boldsymbol{V}_2^\star|\big)^2 + 2|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star|\big(1 - \cos\angle\boldsymbol{V}_{12}^\star\big)\Big\} \end{aligned}$$

を得る。ここで，現実的な潮流状態では，母線の電圧フェーザの位相差である $\angle\boldsymbol{V}_{12}^\star$ は $\pm\frac{\pi}{2}$ の範囲内であることに注意されたい [†1]。このことから，$\boldsymbol{y}$ の実部である送電線のコンダクタンス g が 0 でない場合には，この方程式を満たす電圧フェーザは，必ず

$$|\boldsymbol{V}_1^\star| = |\boldsymbol{V}_2^\star|, \qquad \angle\boldsymbol{V}_{12}^\star = 0 \tag{1.9}$$

を満たさなければならない。しかしながら，式**??**は $P_1^\star$ と $P_2^\star$ がともに 0 であることを意味する。したがって，g が 0 でない限りは，式**??**を満たす定常潮流状態は実現不可能であることが結論づけられる。これは，送電線のコンダクタンス成分（抵抗成分）により送電損失が生じるため，式**??**の設定では電力の需給が系統全体でバランスしないことを表している。このように，「すべての有効電力やすべての無効電力を特定の値に指定してしまうと，式 (**??**) を満たす変数が存在しない場合があること」に注意が必要である。

以下では，簡単化のため，送電線のコンダクタンス g が 0 であると仮定してみよう。また，対地静電容量は十分に小さく，サセプタンス b は負で

[†1] 発電機の偏角差や母線の電圧フェーザの位相差が $\pm\frac{\pi}{2}$ の範囲を超えてしまうと正弦関数の傾きが反転するため，非現実的な有効電力の送電特性となる。

あると仮定する。このとき

$$P_1^\star = -b|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star|\sin\angle\boldsymbol{V}_{12}^\star, \qquad P_2^\star = b|\boldsymbol{V}_1^\star||\boldsymbol{V}_2^\star|\sin\angle\boldsymbol{V}_{12}^\star$$

であることから，電圧フェーザの分布は $P_1^\star$ と $-P_2^\star$ が等しいものでなければならない。例えば，式**??**の値を指定する場合には

$$|\boldsymbol{V}_1^\star| = \sqrt{\frac{2}{|b|}}, \qquad |\boldsymbol{V}_2^\star| = \sqrt{\frac{2}{|b|}}$$

のように電圧フェーザの絶対値を指定することにより，位相差は

$$\angle\boldsymbol{V}_{12}^\star = \frac{\pi}{6}$$

と定められる。既に 3 変数以上の値が定められているため，無効電力は

$$Q_1^\star = 2 - \sqrt{3}, \qquad Q_2^\star = 2 - \sqrt{3}$$

のように自動的に値が定まる。

例**??**で示されたように，与えられた送電網のアドミタンス行列 $\boldsymbol{Y}$ に対して，定常状態における $2N$ 本の連立方程式

$$\begin{cases} P_1^\star + \boldsymbol{j}Q_1^\star = \displaystyle\sum_{j=1}^{N} \overline{\boldsymbol{Y}}_{1j}|\boldsymbol{V}_1^\star||\boldsymbol{V}_j^\star|e^{\boldsymbol{j}(\angle\boldsymbol{V}_1^\star - \angle\boldsymbol{V}_j^\star)} \\ \qquad\qquad \vdots \\ P_N^\star + \boldsymbol{j}Q_N^\star = \displaystyle\sum_{j=1}^{N} \overline{\boldsymbol{Y}}_{Nj}|\boldsymbol{V}_N^\star||\boldsymbol{V}_j^\star|e^{\boldsymbol{j}(\angle\boldsymbol{V}_N^\star - \angle\boldsymbol{V}_j^\star)} \end{cases} \tag{1.10}$$

を満たす，$4N$ 個の定数の組

$$\big(P_1^\star, Q_1^\star, |\boldsymbol{V}_1^\star|, \angle\boldsymbol{V}_1^\star, \ldots, P_N^\star, Q_N^\star, |\boldsymbol{V}_N^\star|, \angle\boldsymbol{V}_N^\star,\big) \tag{1.11}$$

を 1 つ定める手続きが潮流計算である。ただし，電圧フェーザの位相は相対的な値のみが意味をもつため，実質的に定めるべき変数は $(4N-1)$ 個である。**??**

節で上述したように，潮流計算は，系統全体で需要と供給をバランスすることが可能な電力系統モデルの平衡点を求める手続きであると解釈できる。なお，このプロセスにおいては，発電機や負荷などの各機器の特性は考慮されておらず，各母線に対する入出力の定常値のみを求めている。正確には，**??**節におけるステップ B の計算を行うことで，微分代数方程式系の内部状態に関する平衡点が 1 つ定まる。ステップ B の計算は，**??**節で後述する。

1.2.2 定常的な潮流状態の数値的な探索手法

一般に電気学会標準モデル [**?**]，IEEE 39 母線系統モデル [**?**]，IEEE 68 母線系統モデル [**?**] などには，各送電線のインピーダンス値に加えて，各発電機母線に供給される電力や各負荷母線で消費される電力の標準的な値がデータシートとして与えられている。それらの標準的な値に基づき $2N$ 個の変数を指定することによって，残りの変数を数値的に探索することができる。

データシートには，各負荷母線で消費される有効電力と無効電力の値，および，各発電機母線に供給される有効電力の値とその母線における電圧フェーザの絶対値が与えられているのが一般的である。したがって，それらの値を用いることにより，$2N$ 個の定常値をあらかじめ指定することができる。しかしながら，例**??**で示されているように，すべての母線における有効電力の定常値を事前に指定してしまうと，送電損失の影響によって，残りの変数をいかなる値にしても式**??**の連立方程式を満たせなくなってしまう。例えば，一部の負荷母線における有効電力や無効電力の値をデータシートとは異なる定常値に指定すると，発電機母線に供給されるべき有効電力や無効電力の定常値も変化すると同時に，送電網を流れる電力や母線電圧フェーザの定常値も変化するため，系統全体での送電損失の合計値も変化する。したがって，すべての発電機母線における有効電力の定常値を事前に指定してしまうと，式**??**の連立方程式が一般に可解とならない。

この問題を解決するための代表的な方策は，**スラック母線**（slack bus）と呼ばれる特別な発電機母線を 1 つ導入することである。スラック母線では，有効

電力を指定する代わりに，電圧フェーザの位相を指定する。このとき，各母線における電圧フェーザの位相は相対的な値のみが意味をもつため，スラック母線の位相の定常値を 0 に指定しても一般性を失わない。結果として，系統全体での送電損失の値に整合するように，スラック母線における有効電力が自動的に決定される。以上の手順は，つぎのようにまとめられる。

(a) データシートに基づき，スラック母線には $(|\boldsymbol{V}_{i_0}^{\star}|, \angle \boldsymbol{V}_{i_0}^{\star})$ の値，それ以外の発電機母線には $(P_i^{\star}, |\boldsymbol{V}_i^{\star}|)_{i \in \mathcal{I}_{\mathrm{G}} \setminus \{i_0\}}$ の値，負荷母線には $(P_i^{\star}, Q_i^{\star})_{i \in \mathcal{I}_{\mathrm{L}}}$ の値を指定する。

(b) 式**??**の連立方程式を満たすように，その他の変数を数値的に探索する。

ただし，$\mathcal{I}_{\mathrm{G}}$ は発電機母線の添字集合，$\mathcal{I}_{\mathrm{L}}$ は負荷母線の添字集合，$i_0 \in \mathcal{I}_{\mathrm{G}}$ はスラック母線の添字を表す。なお，機器が接続されていない母線は，消費される有効電力と無効電力が 0 である負荷母線として扱う。

表 1.1 データシートと潮流計算結果 (1)

	母線 1	母線 2	母線 3
$P_i^{\star}$	0.5	-3	
$Q_i^{\star}$		0	
$\lvert\boldsymbol{V}_i^{\star}\rvert$	2		2
$\angle\boldsymbol{V}_i^{\star}$			0

(a) データシート

	母線 1	母線 2	母線 3
$P_i^{\star}$			2.5006
$Q_i^{\star}$	0.0157		0.1388
$\lvert\boldsymbol{V}_i^{\star}\rvert$		1.9969	
$\angle\boldsymbol{V}_i^{\star}$	-0.0490	-0.0596	

(b) 潮流計算結果

表 1.2 データシートと潮流計算結果 (2)

	母線 1	母線 2	母線 3
$P_i^{\star}$		-3	0.5
$Q_i^{\star}$		0	
$\lvert\boldsymbol{V}_i^{\star}\rvert$	2		2
$\angle\boldsymbol{V}_i$	0		

(a) データシート

	母線 1	母線 2	母線 3
$P_i^{\star}$	2.5158		
$Q_i^{\star}$	-0.0347		0.1759
$\lvert\boldsymbol{V}_i^{\star}\rvert$		1.9918	
$\angle\boldsymbol{V}_i^{\star}$		-0.0538	-0.0419

(b) 潮流計算結果

例 1.2 (データシートに基づく潮流計算) 例**??**で議論した 3 母線で構成される電力系統モデルを考えよう。式**??**の送電網のアドミタンス行列に対して，2 つの送電線のアドミタンス値は

$$\boldsymbol{y}_{12} = 1.3652 - \boldsymbol{j}11.6041, \qquad \boldsymbol{y}_{23} = -\boldsymbol{j}10.5107 \tag{1.12}$$

に設定する。母線 2 と母線 3 を結ぶ送電線のコンダクタンス（$\boldsymbol{y}_{23}$ の実部）は 0 に設定されており，その送電線における有効電力の送電損失は 0 となることに注意されたい。

まず，母線 1 が発電機母線，母線 2 が負荷母線，母線 3 がスラック母線である場合を考える。具体的には，**??**(a) に示されている値が各母線に指定されているものとする。このとき，式**??**の連立方程式を満たす変数の組は，**??**(b) のように求められる。この場合の送電損失は

$$P_1^\star + P_2^\star + P_3^\star = 6.2562 \times 10^{-4}, \qquad Q_1^\star + Q_2^\star + Q_3^\star = 1.5450 \times 10^{-1}$$

となる。負荷の消費電力に対する送電損失の比率は 0.02%程度と非常に小さい。有効電力の送電損失が小さい理由は，母線 2 の負荷で消費される有効電力が 3 [pu] であるのに対して，その大半の約 2.5 [pu] を母線 3 の発電機から供給しており，母線 1 の発電機からは 0.5 [pu] しか供給していないためである。すなわち，母線 2 で消費される有効電力の大半を送電損失が生じない右側の送電線を用いて供給している。

つぎに，母線 1 がスラック母線，母線 2 が負荷母線，母線 3 が発電機母線である場合として，**??**(a) のように各母線の変数を指定する。このときの潮流計算の結果は**??**(b) となる。この場合の送電損失は

$$P_1^\star + P_2^\star + P_3^\star = 1.5826 \times 10^{-2}, \qquad Q_1^\star + Q_2^\star + Q_3^\star = 1.4120 \times 10^{-1}$$

となる。この場合の負荷の消費電力に対する送電損失の比率は約 0.52%である。先の例と比較して，有効電力の送電損失が大きくなっていることが

わかる。これは，送電損失が生じる左側の送電線を使って，母線 2 で消費される有効電力の大半を供給したことに起因する。一方で，無効電力の損失は，先の例と比較して小さくなっていることもわかる。

例**??**から，求められる潮流状態によって，系統全体の送電損失の大きさが異なることがわかる。一般に，コンダクタンス成分（抵抗成分）が大きい送電線を使って消費電力を供給する場合に，有効電力の送電損失が大きくなる。送電損失が大きくなるほど，同じ有効電力の消費量を供給するために必要な発電量が大きくなるため，発電費用などの経済コストも大きくなってしまう。

一方で，経済コストの低い定常潮流状態が，必ずしも安定度の高い平衡点ではない点にも注意が必要である。したがって，経済性や安定度などに関するトレードオフを考慮して，より良い平衡点を探索することが実応用では重要となる。そのようなより良い平衡点探索のプロセスは，電力系統工学では**最適潮流計算**（optimal power flow calculation）と呼ばれる。本書では，平衡点の選び方と安定度の関係を**??**章や**??**章で議論する。

1.2.3 アドミタンス行列と送電損失の関係 *

以下では，N 個の母線で構成される一般的な送電網に対して，任意の潮流状態に関する送電損失の数学的表現を導出する。アドミタンス行列 $\boldsymbol{Y}$ の実部と虚部であるコンダクタンス行列 G とサセプタンス行列 B がそれぞれ対称である場合には，適当な定数 $\phi_{ij} = \phi_{ji}$，$\psi_{ij} = \psi_{ji}$ を用いて

$$G_{ij} = \begin{cases} \sum_{j=1}^{N} \phi_{ij}, & i = j \\ -\phi_{ij}, & i \neq j \end{cases} \qquad B_{ij} = \begin{cases} -\sum_{j=1}^{N} \psi_{ij}, & i = j \\ \psi_{ij}, & i \neq j \end{cases} \tag{1.13}$$

の形式で書き表すことができる。ただし，G_{ij} と B_{ij} はそれぞれ，コンダクタンス行列 G とサセプタンス行列 B の第 (i, j) 要素を表す。式**??**は

$$\phi_{ii} := \sum_{j=1}^{N} G_{ij}, \qquad \phi_{ij} := -G_{ij}, \qquad \psi_{ii} := -\sum_{j=1}^{N} B_{ij}, \qquad \psi_{ij} := B_{ij}$$

と定義していることに等しい。この表現を用いるとつぎの事実が示される。

定理 1.1 (送電損失の母線電圧フェーザによる表現) 式??に対して，系統全体での有効電力と無効電力の送電損失として

$$L_P(t) := P_1(t) + \cdots P_N(t), \qquad L_Q(t) := Q_1(t) + \cdots Q_N(t) \quad (1.14)$$

を定義する。これらの送電損失は

$$\begin{aligned} L_P(t) &= \sum_{i=1}^{N} \phi_{ii} |\boldsymbol{V}_i(t)|^2 + \sum_{i=1}^{N} \sum_{j=i+1}^{N} \phi_{ij} W\big(\boldsymbol{V}_i(t), \boldsymbol{V}_j(t)\big) \\ L_Q(t) &= \sum_{i=1}^{N} \psi_{ii} |\boldsymbol{V}_i(t)|^2 + \sum_{i=1}^{N} \sum_{j=i+1}^{N} \psi_{ij} W\big(\boldsymbol{V}_i(t), \boldsymbol{V}_j(t)\big) \end{aligned} \quad (1.15)$$

で与えられる。ただし

$$W(\boldsymbol{V}_i, \boldsymbol{V}_j) := \big(|\boldsymbol{V}_i| - |\boldsymbol{V}_j|\big)^2 + 2|\boldsymbol{V}_i||\boldsymbol{V}_j|\big\{1 - \cos(\angle \boldsymbol{V}_i - \angle \boldsymbol{V}_j)\big\}$$

とする。

証明 表記の簡単化のため，時刻 t は省略する。また，$\angle \boldsymbol{V}_i - \angle \boldsymbol{V}_j$ を $\angle \boldsymbol{V}_{ij}$ と表す。式??より，$\boldsymbol{Y}_{ij} = G_{ij} + \boldsymbol{j} B_{ij}$ に対して，$\boldsymbol{Y}_{ij} = \boldsymbol{Y}_{ji}$ より

$$\begin{aligned} \sum_{i=1}^{N} (P_i + \boldsymbol{j} Q_i) &= \sum_{i=1}^{N} \overline{\boldsymbol{Y}}_{ii} |\boldsymbol{V}_i|^2 + \sum_{i=1}^{N} \sum_{j=i+1}^{N} \overline{\boldsymbol{Y}}_{ij} |\boldsymbol{V}_i||\boldsymbol{V}_j| \left(e^{\boldsymbol{j}\angle \boldsymbol{V}_{ij}} + e^{\boldsymbol{j}\angle \boldsymbol{V}_{ji}} \right) \\ &= \sum_{i=1}^{N} \overline{\boldsymbol{Y}}_{ii} |\boldsymbol{V}_i|^2 + 2 \sum_{i=1}^{N} \sum_{j=i+1}^{N} \overline{\boldsymbol{Y}}_{ij} |\boldsymbol{V}_i||\boldsymbol{V}_j| \cos \angle \boldsymbol{V}_{ij} \end{aligned}$$

が成り立つ。したがって，式??の G_{ij} と B_{ij} の表記を用いれば

$$\begin{aligned} L_P &= \sum_{i=1}^{N} \left(\sum_{j=1}^{N} \phi_{ij} \right) |\boldsymbol{V}_i|^2 - 2 \sum_{i=1}^{N} \sum_{j=i+1}^{N} \phi_{ij} |\boldsymbol{V}_i||\boldsymbol{V}_j| \cos \angle \boldsymbol{V}_{ij} \\ L_Q &= \sum_{i=1}^{N} \left(\sum_{j=1}^{N} \psi_{ij} \right) |\boldsymbol{V}_i|^2 - 2 \sum_{i=1}^{N} \sum_{j=i+1}^{N} \psi_{ij} |\boldsymbol{V}_i||\boldsymbol{V}_j| \cos \angle \boldsymbol{V}_{ij} \end{aligned}$$

が得られる。ここで，L_P の第 1 項に注目すると，$\phi_{ij} = \phi_{ji}$ であることから

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\phi_{ij}|\boldsymbol{V}_i|^2=\sum_{i=1}^{N}\phi_{ii}|\boldsymbol{V}_i|^2+\sum_{i=1}^{N}\sum_{j=i+1}^{N}\phi_{ij}|\boldsymbol{V}_i|^2+\sum_{i=1}^{N}\sum_{j=i+1}^{N}\phi_{ij}|\boldsymbol{V}_j|^2$$

がわかる。これを用いて L_P の第 1 項を書き換えれば

$$\begin{aligned}L_P&=\sum_{i=1}^{N}\phi_{ii}|\boldsymbol{V}_i|^2+\sum_{i=1}^{N}\sum_{j=i+1}^{N}\phi_{ij}\left(|\boldsymbol{V}_i|^2+|\boldsymbol{V}_j|^2-2|\boldsymbol{V}_i||\boldsymbol{V}_j|\cos\angle\boldsymbol{V}_{ij}\right)\\&=\sum_{i=1}^{N}\phi_{ii}|\boldsymbol{V}_i|^2+\sum_{i=1}^{N}\sum_{j=i+1}^{N}\phi_{ij}\left\{(|\boldsymbol{V}_i|-|\boldsymbol{V}_j|)^2+2|\boldsymbol{V}_i||\boldsymbol{V}_j|(1-\cos\angle\boldsymbol{V}_{ij})\right\}\end{aligned}$$

が得られる。したがって，L_P は式**??**の形式で表せることがわかる。同様の手順により，L_Q は式**??**の形式で表せることもわかる。 □

定理**??**は，例**??**で示されている 2 母線の電力系統における電力損失の議論が，任意の個数の母線から構成される電力系統にも同様に一般化できることを示している。ここで，**??**節のようにアドミタンス行列 $\boldsymbol{Y}$ を

$$\boldsymbol{Y}=\boldsymbol{Y}_0+\boldsymbol{j}\,\mathsf{diag}(b_i)_{i\in\{1,\ldots,N\}}$$

と表す場合には，$\boldsymbol{Y}_0\mathbb{1}$ は 0 であること，すなわち，$\boldsymbol{Y}_0$ の実部と虚部に対して各々すべての行の和は 0 であることから

$$\phi_{ii}=0,\qquad \psi_{ii}=-b_i$$

となる。また，各送電線のコンダクタンスは非負であり，サセプタンスは負であることから，すべての $i\neq j$ に対して，ϕ_{ij} と ψ_{ij} は非負である。したがって，ϕ_{ij} が 0 でない限りは，母線 i と母線 j の間で有効電力を送電する場合に必ず損失が生じることがわかる。これにより，系統全体での有効電力の送電損失 $L_P(t)$ は，任意の時刻 t で正であることがわかる。同様に，b_i が十分に小さい場合，すなわち，送電線の対地静電容量が十分に小さい場合には，無効電力の損失 $L_Q(t)$ も正であることがわかる。言い換えれば，キャパシタンスの特性をもつ機器を母線に接続することは，無効電力の損失を減らす効果を与える。

1.3 所与の潮流状態を実現する各機器のパラメータ設定

本節では，**??**節のステップ B として説明された，潮流計算の結果に整合するように発電機の内部状態の定常値や外部入力値，負荷のパラメータ値を逆算する方法を説明する。

1.3.1 所望の電力供給を実現する発電機の定常状態

??節における電圧フェーザを入力とする発電機モデルを考えよう。表記の簡単化のため，添字 i を省略して

$$\left\{\begin{array}{r} \dot{\delta} = \omega_0 \Delta\omega \\ M\Delta\dot{\omega} = -D\Delta\omega - P + P_{\rm mech} \\ \tau\dot{E} = -\frac{X}{X'}E + \left(\frac{X}{X'} - 1\right)|\boldsymbol{V}|\cos(\delta - \angle\boldsymbol{V}) + V_{\rm field} \end{array}\right. \tag{1.16}$$

と表す。ここで，有効電力と無効電力を出力とする場合には

$$P = \frac{|\boldsymbol{V}|E}{X'}\sin(\delta - \angle\boldsymbol{V}), \qquad Q = \frac{|\boldsymbol{V}|E}{X'}\cos(\delta - \angle\boldsymbol{V}) - \frac{|\boldsymbol{V}|^2}{X'} \tag{1.17}$$

である。ここでの目的は，潮流計算の結果として，発電機が接続される母線の有効電力，無効電力，電圧フェーザの絶対値と位相が定数で与えられた場合に，それらの値に整合する発電機の内部状態と外部入力値の定常値を求めることである。具体的には，与えられた有効電力，無効電力，電圧フェーザの絶対値と位相の組を $(P^\star, Q^\star, |V^\star|, \angle\boldsymbol{V}^\star)$ と表すとき

$$\left\{\begin{array}{r} P^\star = \frac{|V^\star|E^\star}{X'}\sin(\delta^\star - \angle\boldsymbol{V}^\star), \\ Q^\star = \frac{|V^\star|E^\star}{X'}\cos(\delta^\star - \angle\boldsymbol{V}^\star) - \frac{|V^\star|^2}{X'}, \\ 0 = -P^\star + P^\star_{\rm mech}, \\ 0 = -\frac{X}{X'}E^\star + \left(\frac{X}{X'} - 1\right)|V^\star|\cos(\delta^\star - \angle\boldsymbol{V}^\star) + V^\star_{\rm field} \end{array}\right. \tag{1.18}$$

の連立方程式を満たす発電機の内部状態の定常値 $(\delta^\star, E^\star)$ と外部入力の定常値

$(P^\star_{\rm mech}, V^\star_{\rm field})$ を求めることが目的となる。式**??**の連立方程式は，式**??**において角周波数偏差 $\Delta\omega$ の定常値が 0 であるとした場合の平衡点に関する方程式である。また，与えられた $(P^\star, Q^\star, |V^\star|, \angle \boldsymbol{V}^\star)$ は，各母線に対する発電機モデルの入出力値に対応している。

式**??**を満たす発電機の内部状態の定常値を具体的に計算すると

$$\begin{aligned} \delta^\star &= \angle \boldsymbol{V}^\star + \arctan\left(\frac{P^\star}{Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X'}}\right), \\ E^\star &= \frac{X'}{|\boldsymbol{V}^\star|}\sqrt{\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X'}\right)^2 + (P^\star)^2} \end{aligned} \tag{1.19a}$$

となる。また，機械入力と界磁電圧の定常値は

$$\begin{aligned} P^\star_{\rm mech} &= P^\star, \\ V^\star_{\rm field} &= \frac{\frac{X}{|\boldsymbol{V}^\star|}\left\{\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X'}\right)\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X}\right) + (P^\star)^2\right\}}{\sqrt{\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X'}\right)^2 + (P^\star)^2}} \end{aligned} \tag{1.19b}$$

となる。これらの導出過程は，**??**節を参照されたい。

1.3.2 所望の電力消費を実現する負荷のパラメータ

??節で説明された負荷モデルに対して，所望の電力消費に整合するパラメータを設定する方法を説明する。式**??**を用いて電流フェーザを消去すると，定インピーダンスモデルは

$$P + \boldsymbol{j}Q = -\frac{|\boldsymbol{V}|^2}{\overline{\boldsymbol{z}}^\star_{\rm load}} \tag{1.20}$$

と書き表される。ただし，母線の添字 i は表記の簡単化のため省略した。これは電流フェーザ $\boldsymbol{V}$ を入力，有効電力 P と無効電力 Q を出力とした場合の負荷の定インピーダンスモデルと解釈できる。潮流計算により定められた有効電力と無効電力の値を $P^\star$ と $Q^\star$，電圧フェーザの絶対値を $|\boldsymbol{V}^\star|$ と表せば，負荷のインピーダンス $\boldsymbol{z}^\star_{\rm load}$ の実部（抵抗）と虚部（リアクタンス）は

$$\mathsf{Re}[\boldsymbol{z}^{\star}_{\mathrm{load}}] = -\frac{P^{\star}|\boldsymbol{V}^{\star}|^2}{(P^{\star})^2+(Q^{\star})^2}, \qquad \mathsf{Im}[\boldsymbol{z}^{\star}_{\mathrm{load}}] = -\frac{Q^{\star}|\boldsymbol{V}^{\star}|^2}{(P^{\star})^2+(Q^{\star})^2} \quad (1.21)$$

と求められる。同様に，定電流モデルは

$$P + \boldsymbol{j}Q = \overline{\boldsymbol{I}}^{\star}_{\mathrm{load}}|\boldsymbol{V}| \quad (1.22)$$

と書き表されることから，負荷の電流パラメータの実部と虚部はそれぞれ

$$\mathsf{Re}[\boldsymbol{I}^{\star}_{\mathrm{load}}] = \frac{P^{\star}}{|\boldsymbol{V}^{\star}|}, \qquad \mathsf{Im}[\boldsymbol{I}^{\star}_{\mathrm{load}}] = -\frac{Q^{\star}}{|\boldsymbol{V}^{\star}|}$$

と求められる。定電力モデルは

$$P + \boldsymbol{j}Q = P^{\star}_{\mathrm{load}} + \boldsymbol{j}Q^{\star}_{\mathrm{load}} \quad (1.23)$$

であることから，明らかにそのパラメータは

$$P^{\star}_{\mathrm{load}} = P^{\star}, \qquad Q^{\star}_{\mathrm{load}} = Q^{\star}$$

である。これらのパラメータ値を負荷モデルに設定すれば，潮流計算によって求められた潮流状態が定常的に実現される。

1.3.3 発電機の内部状態と入出力の数学的関係 ⋆

以下では，発電機の内部状態，母線に供給される有効電力と無効電力，母線の電圧フェーザの間に成り立つ関係を数学的に解析する。ここでは，**??**節で扱った発電機モデルを用いる。ただし，表記の簡単化のため，添字 i を省略して

$$\begin{cases} \dot{\delta} = \omega_0 \Delta\omega \\ M\Delta\dot{\omega} = -D\Delta\omega - P + P_{\mathrm{mech}} \\ \tau\dot{E} = -\frac{X_{\mathrm{d}}}{X'_{\mathrm{d}}}E + \left(\frac{X_{\mathrm{q}}}{X'_{\mathrm{d}}} - 1\right)|\boldsymbol{V}|\cos(\delta - \angle\boldsymbol{V}) + V_{\mathrm{field}} \end{cases} \quad (1.24)$$

とする。ここで，有効電力と無効電力を出力とする場合には

$$\begin{aligned} P &= \frac{|\boldsymbol{V}|E}{X'_{\mathrm{d}}}\sin(\delta - \angle\boldsymbol{V}) - \left(\frac{1}{X'_{\mathrm{d}}} - \frac{1}{X_{\mathrm{q}}}\right)|\boldsymbol{V}|^2\sin(\delta - \angle\boldsymbol{V})\cos(\delta - \angle\boldsymbol{V}), \\ Q &= \frac{|\boldsymbol{V}|E}{X'_{\mathrm{d}}}\cos(\delta - \angle\boldsymbol{V}) - |\boldsymbol{V}|^2\left(\frac{\cos^2(\delta - \angle\boldsymbol{V})}{X'_{\mathrm{d}}} + \frac{\sin^2(\delta - \angle\boldsymbol{V})}{X_{\mathrm{q}}}\right) \end{aligned} \quad (1.25)$$

である。なお，X'_{d} と X_{q} が等しく X' であり，X_{d} を X に置き換えれば，**??**節で扱った発電機モデルに一致する。また，電流フェーザを出力とする場合には

$$\begin{aligned} |\boldsymbol{I}|\cos(\delta-\angle\boldsymbol{I}) &= \frac{|\boldsymbol{V}|}{X_{\mathrm{q}}}\sin(\delta-\angle\boldsymbol{V}),\\ |\boldsymbol{I}|\sin(\delta-\angle\boldsymbol{I}) &= \frac{E-|\boldsymbol{V}|\cos(\delta-\angle\boldsymbol{V})}{X'_{\mathrm{d}}} \end{aligned} \tag{1.26}$$

である。なお，式**??**と式**??**は等価な出力であること，すなわち，与えられた任意の $(\delta, E, |\boldsymbol{V}|, \angle\boldsymbol{V})$ に対して，(P, Q) と $(|\boldsymbol{I}|, \angle\boldsymbol{I})$ には一対一の関係が存在することに注意されたい。

この発電機モデルに対して，つぎの事実が示される。

補題 1.1 (発電機の内部状態と入出力の関係)　式**??**を $\delta-\angle\boldsymbol{V}$ と E に関する連立方程式と考えるとき，その解は

$$\delta-\angle\boldsymbol{V} = \arctan\left(\frac{P}{Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}}\right), \tag{1.27a}$$

$$E = \frac{\frac{X'_{\mathrm{d}}}{|\boldsymbol{V}|}\left\{\left(Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}\right)\left(Q+\frac{|\boldsymbol{V}|^2}{X'_{\mathrm{d}}}\right)+P^2\right\}}{\sqrt{\left(Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}\right)^2+P^2}} \tag{1.27b}$$

で与えられる。ただし，$|\boldsymbol{V}|\neq 0$ とする。逆に，式 (**??**) を P と Q に関する連立方程式と考えるとき，その解は式**??**で与えられる。

証明　まず，式**??**から式 (**??**) を導く。式**??**の P に $\cos(\delta-\angle\boldsymbol{V})$ を乗じ，Q に $\sin(\delta-\angle\boldsymbol{V})$ を乗じて差をとれば

$$P\cos(\delta-\angle\boldsymbol{V}) - Q\sin(\delta-\angle\boldsymbol{V}) = \frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}\sin(\delta-\angle\boldsymbol{V})$$

が得られる。この両辺を $\cos(\delta-\angle\boldsymbol{V})$ で割れば，式**??**の関係が得られる。つぎに，式**??**の関係を示す。式**??**を用いて P と Q を $\boldsymbol{I}$ で書き直すと，式**??**は式**??**に等価変形される。また，これは

$$|\boldsymbol{V}|e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})} = E - X'_{\mathrm{d}}|\boldsymbol{I}|\sin(\delta-\angle\boldsymbol{I}) + \boldsymbol{j}X_{\mathrm{q}}|\boldsymbol{I}|\cos(\delta-\angle\boldsymbol{I}) \tag{1.28}$$

と等価である。式**??**の関係を複素数で表現すると

$$\frac{e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})}-e^{-\boldsymbol{j}(\delta-\angle\boldsymbol{V})}}{e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})}+e^{-\boldsymbol{j}(\delta-\angle\boldsymbol{V})}}=\underbrace{\frac{P}{Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}}}_{\alpha}\boldsymbol{j}$$

であることから

$$e^{-\boldsymbol{j}(\delta-\angle\boldsymbol{V})}=\frac{1-\alpha\boldsymbol{j}}{1+\alpha\boldsymbol{j}}e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})}$$

がわかる。したがって，式**??**を等価変形した

$$|\boldsymbol{I}|e^{\boldsymbol{j}(\delta-\angle\boldsymbol{I})}=\frac{P+\boldsymbol{j}Q}{|\boldsymbol{V}|}e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})} \tag{1.29}$$

について，その複素共役を考えることにより

$$|\boldsymbol{I}|e^{-\boldsymbol{j}(\delta-\angle\boldsymbol{I})}=\frac{P-\boldsymbol{j}Q}{|\boldsymbol{V}|}\cdot\frac{1-\alpha\boldsymbol{j}}{1+\alpha\boldsymbol{j}}e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})} \tag{1.30}$$

を得る。式**??**と式**??**から

$$\begin{aligned}|\boldsymbol{I}|\sin(\delta-\angle\boldsymbol{I})&=\frac{1}{|\boldsymbol{V}|}\cdot\frac{\alpha P+Q}{1+\alpha\boldsymbol{j}}e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})},\\ |\boldsymbol{I}|\cos(\delta-\angle\boldsymbol{I})&=\frac{1}{|\boldsymbol{V}|}\cdot\frac{P-\alpha Q}{1+\alpha\boldsymbol{j}}e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})}\end{aligned}$$

がわかる。これらを式**??**に代入することによって，$\boldsymbol{I}$ を P と Q で改めて書き直せば，式**??**の関係が成り立つとき，式**??**が

$$E=\frac{X'_{\mathrm{d}}}{|\boldsymbol{V}|}\left\{\left(Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}\right)\left(Q+\frac{|\boldsymbol{V}|^2}{X'_{\mathrm{d}}}\right)+P^2\right\}\frac{Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}-\boldsymbol{j}P}{\left(Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}\right)^2+P^2}e^{\boldsymbol{j}(\delta-\angle\boldsymbol{V})} \tag{1.31}$$

と等価であることがわかる。ここで，式**??**の関係から

$$Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}-\boldsymbol{j}P=\left|Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}-\boldsymbol{j}P\right|e^{-\boldsymbol{j}(\delta-\angle\boldsymbol{V})}$$

が成り立つことがわかる。さらに

$$|E|=E,\qquad\left(Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}\right)^2+P^2=\left|Q+\frac{|\boldsymbol{V}|^2}{X_{\mathrm{q}}}-\boldsymbol{j}P\right|^2$$

であることから，式**??**の関係が得られる。

逆の手順をたどって，式 (**??**) から式**??**を導く。式**??**の関係を用いると，式**??**の E は式**??**の E で書き換えられる。前述のように，式**??**の関係が成り立つとき，式**??**は式**??**と等価である。 □

補題**??**は，変数の組 $(\delta - \angle \boldsymbol{V}, E)$ と組 (P, Q) の間に一対一の関係があることを示している。特に，発電機の入出力である $(|\boldsymbol{V}|, \angle \boldsymbol{V})$ や (P, Q) から，発電機の内部状態である (δ, E) を一意的に逆算できることを示している。なお，式 (**??**) の関係は定常状態，過渡状態に関わらず任意の時刻 t で成り立つことに注意されたい。

つぎの定理は，発電機の定常状態において，入力，出力，内部状態の間に成り立つ関係を与える。

定理 1.2 (定常状態における発電機の内部状態と入出力の関係) 式**??**および式**??**の発電機モデルを考える。ある実定数 $|\boldsymbol{V}^\star|$，$\Delta\omega^\star$，$\angle \boldsymbol{V}^\star$，$P^\star$，$Q^\star$ に対して，機械入力と界磁電圧による入力を

$$P_{\mathrm{mech}}(t) = D\Delta\omega^\star + P^\star,$$
$$V_{\mathrm{field}}(t) = \frac{\frac{X_{\mathrm{d}}}{|\boldsymbol{V}^\star|}\left\{\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X_{\mathrm{q}}}\right)\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X_{\mathrm{d}}}\right) + (P^\star)^2\right\}}{\sqrt{\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X_{\mathrm{q}}}\right)^2 + (P^\star)^2}} \tag{1.32a}$$

で与えられる定数とし，母線の電圧フェーザによる入力が

$$|\boldsymbol{V}(t)| = |\boldsymbol{V}^\star|, \qquad \angle \boldsymbol{V}(t) = \omega_0 \Delta\omega^\star t + \angle \boldsymbol{V}^\star \tag{1.32b}$$

と定められているものとする。このとき，回転子偏角，角周波数偏差，内部電圧は

$$
\begin{aligned}
\delta(t) &= \angle \boldsymbol{V}(t) + \arctan\left(\frac{P^\star}{Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X_\mathrm{q}}}\right), \\
\Delta\omega(t) &= \Delta\omega^\star, \\
E(t) &= \frac{\frac{X'_\mathrm{d}}{|\boldsymbol{V}^\star|}\left\{\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X_\mathrm{q}}\right)\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X'_\mathrm{d}}\right) + (P^\star)^2\right\}}{\sqrt{\left(Q^\star + \frac{|\boldsymbol{V}^\star|^2}{X_\mathrm{q}}\right)^2 + (P^\star)^2}}
\end{aligned}
\tag{1.33}
$$

を定常解にもつ。また，母線に供給される有効電力と無効電力は

$$
P(t) = P^\star, \qquad Q(t) = Q^\star \tag{1.34}
$$

で与えられる定数となる。

証明 まず，式 (??) の入力のもとで，式??が式??の微分方程式の解であることを仮定した場合に，出力に関して式??が成り立つことを示す。補題??で示されているように，式??を $P^\star$ と $Q^\star$ に関する方程式と考えれば，それらの解は

$$
\begin{aligned}
P^\star &= \frac{|\boldsymbol{V}^\star|E(t)}{X'_\mathrm{d}}\sin\left(\delta(t) - \angle\boldsymbol{V}(t)\right) \\
&\quad - \left(\frac{1}{X'_\mathrm{d}} - \frac{1}{X_\mathrm{q}}\right)|\boldsymbol{V}^\star|^2 \sin\left(\delta(t) - \angle\boldsymbol{V}(t)\right)\cos\left(\delta(t) - \angle\boldsymbol{V}(t)\right), \\
Q^\star &= \frac{|\boldsymbol{V}^\star|E(t)}{X'_\mathrm{d}}\cos\left(\delta(t) - \angle\boldsymbol{V}(t)\right) \\
&\quad - |\boldsymbol{V}^\star|^2\left(\frac{\cos^2\left(\delta(t) - \angle\boldsymbol{V}(t)\right)}{X'_\mathrm{d}} + \frac{\sin^2\left(\delta(t) - \angle\boldsymbol{V}(t)\right)}{X_\mathrm{q}}\right)
\end{aligned}
$$

で与えられる。これは式??を意味している。

つぎに，式 (??) の入力のもとで，式??が式??の微分方程式の解であることを確かめる。式??の δ と $\Delta\omega$ に関する微分方程式は

$$
\frac{M}{\omega_0}\ddot{\delta}(t) + \frac{D}{\omega_0}\dot{\delta}(t) + P(t) - P_\mathrm{mech}(t) = 0
$$

と等価である。式??の $P_\mathrm{mech}(t)$，式??の $P(t)$，式??の $\delta(t)$ を代入すれば，式??の関係から，この微分方程式が満たされることがわかる。同様に，式??の $V_\mathrm{field}(t)$，式??の $\delta(t) - \angle\boldsymbol{V}(t)$ と $E(t)$ を代入することにより，式??の E に関する微分方程式が満たされることがわかる。ただし

$$\cos\left(\arctan\left(\frac{P^\star}{Q^\star+\frac{|\boldsymbol{V}^\star|^2}{X_\mathrm{q}}}\right)\right)=\frac{Q^\star+\frac{|\boldsymbol{V}^\star|^2}{X_\mathrm{q}}}{\sqrt{\left(Q^\star+\frac{|\boldsymbol{V}^\star|^2}{X_\mathrm{q}}\right)^2+(P^\star)^2}}$$

であることを用いる。以上より，求める結果がしたがう。 □

定理**??**から，潮流計算で定められた母線の電圧フェーザ，有効電力，無効電力を実現するために必要な機械入力と界磁電圧の値 $(P^\star_\mathrm{mech}, V^\star_\mathrm{field})$，および，そのときの内部状態 (δ, E) の定常的な挙動を知ることができる。なお，式**??**では電圧フェーザの位相が定数となっていないが，角周波数偏差の定常値を表す $\Delta\omega^\star$ は，通常は 0 に設定するべき定数であるため，実用上の意味をもつのは潮流計算で定められる $\angle\boldsymbol{V}^\star$ の値のみである。明らかに，$\Delta\omega^\star$ が 0 のとき

$$P_\mathrm{mech}(t)=P^\star,\qquad \delta(t)=\angle\boldsymbol{V}^\star+\arctan\left(\frac{P^\star}{Q^\star+\frac{|\boldsymbol{V}^\star|^2}{X_\mathrm{q}}}\right)$$

である。

以上の議論から，発電機の動特性を考慮せず $(P^\star, Q^\star, |\boldsymbol{V}^\star|, \angle\boldsymbol{V}^\star)$ を潮流計算で定めたとしても，それらに整合するような $(P^\star_\mathrm{mech}, V^\star_\mathrm{field})$ を一意的に逆算できることがわかる。この結果から，式 **(??)** を導くことができる。

さらに，つぎの定理は，発電機の定常状態において成り立つ，入出力と内部状態に関する等価関係を与える。

定理 1.3　(発電機の入出力と内部状態に関する等価関係)　式**??**および式**??**の発電機モデルに対して

$$\frac{d^2\delta}{dt^2}(t)=0,\quad \frac{dE}{dt}(t)=0,\quad \frac{dP_\mathrm{mech}}{dt}(t)=0,\quad \frac{dV_\mathrm{field}}{dt}(t)=0 \quad (1.35)$$

がすべての $t\geqq 0$ に対して成り立つための必要十分条件は

$$\frac{dP}{dt}(t)=0,\quad \frac{dQ}{dt}(t)=0,\quad \frac{d|\boldsymbol{V}|}{dt}(t)=0,\quad \frac{d^2\angle\boldsymbol{V}}{dt^2}(t)=0 \quad (1.36)$$

がすべての $t\geqq 0$ に対して成り立つことである。また，式**??**または式**??**が成り立つとき

$$\Delta\omega(t) = \frac{1}{\omega_0}\frac{d\angle \boldsymbol{V}}{dt}(t) \tag{1.37}$$

であり，これは定数である。

証明 まず, 式**??**が成り立つならば, 式**??**が成り立つことを示す。式**??**において, $\Delta\omega$, E, $P_{\rm mech}$, $V_{\rm field}$ はすべて定数であることから, P と $|\boldsymbol{V}|\cos(\delta - \angle\boldsymbol{V})$ が定数であることがわかる。したがって, 式**??**の 2 つの方程式から, Q と $|\boldsymbol{V}|\sin(\delta - \angle\boldsymbol{V})$ も定数であることがわかる。また

$$|\boldsymbol{V}|^2\cos^2(\delta - \angle\boldsymbol{V}) + |\boldsymbol{V}|^2\sin^2(\delta - \angle\boldsymbol{V}) = |\boldsymbol{V}|^2$$

であり，左辺が定数であることから $|\boldsymbol{V}|$ も定数である。さらに，式 (**??**) の第 1 式の関係において，右辺は定数であることから，$\angle\boldsymbol{V}$ と δ の導関数は任意の次数で等しい。したがって

$$\frac{d^2\angle\boldsymbol{V}}{dt^2} = \frac{d^2\delta}{dt^2} = 0$$

が得られる。つぎに, 式**??**が成り立つならば, 式**??**が成り立つことを示す。式 (**??**) から, E が定数であること, および, δ の 2 次導関数が 0 であること, すなわち, $\Delta\omega$ が定数であることがわかる。したがって，式**??**において，P と $|\boldsymbol{V}|\cos(\delta - \angle\boldsymbol{V})$ が定数であることから，$P_{\rm mech}$ と $V_{\rm field}$ が定数であることがわかる。式**??**は $\angle\boldsymbol{V}$ と δ の導関数が等しいことから明らかである。 □

定理**??**が示すように, 発電機への外部入力 $(P_{\rm mech}, V_{\rm field})$ が定数であり, かつ, 内部状態 (δ, E) が定常状態にあることと，母線に対する入出力 $(P, Q, |\boldsymbol{V}|, \angle\boldsymbol{V})$ が定常状態にあることは等価である。したがって，各母線の有効電力や無効電力，電圧フェーザを決定する潮流計算の手続きは，すべての発電機の内部状態と外部入力が定常状態にあると仮定して，電力系統全体の定常状態，すなわち，電力系統モデルの平衡点の 1 つを探索することと数学的に等価である。

1.4 電力系統モデルの時間応答計算

本節では，**??**節のステップ C として説明された電力系統モデルの時間応答の

計算方法を説明する。また，いくつかの外乱に対する電力系統モデルの振る舞いを数値的に解析する。

1.4.1 初 期 値 応 答

??節と**??**節で示された手順によって，すべての発電機の角周波数偏差が 0 となるような定常状態として，電力系統モデルの平衡点が 1 つ求められる。具体的には，潮流計算で定められた母線変数を用いて，各発電機モデルには定理**??**で示される内部状態の初期値と外部入力の定常値を設定し，各負荷モデルには**??**節で逆算された定数を設定すれば，微分代数方程式系で表される電力系統モデルは所与の定常潮流状態で平衡する。特に，求められた平衡点が適切な意味で安定であれば，発電機の内部状態に対して何らかの摂動が生じたとしても，電力系統モデルの内部状態はもとの定常値に漸近的に収束する。この事実を以下の例で確認してみよう。

表 1.3 ??の潮流計算結果に対する発電機の定常値

i	$P^{\star}_{\mathrm{mech}i}$ [pu]	$V^{\star}_{\mathrm{filed}i}$ [pu]	$\delta^{\star}_i$ [rad]	$\Delta\omega^{\star}_i$ [pu]	$E^{\star}_i$ [pu]
1	0.5000	2.0442	0.0670	0	2.0210
3	2.5006	2.5062	0.3870	0	2.2097

表 1.4 ??の潮流計算結果に対する発電機の定常値

i	$P^{\star}_{\mathrm{mech}i}$ [pu]	$V^{\star}_{\mathrm{filed}i}$ [pu]	$\delta^{\star}_i$ [rad]	$\Delta\omega^{\star}_i$ [pu]	$E^{\star}_i$ [pu]
1	2.5158	2.7038	0.5356	0	2.3069
3	0.5000	2.1250	0.0390	0	2.0654

例 1.3 (電力系統モデルの初期値応答) 例**??**で扱った 3 つの母線で構成される電力系統モデルを考えよう。母線 1 と母線 3 には，式**??**の発電機モデルが接続されており，母線 2 には，式**??**の定インピーダンスの負荷モデルが接続されている場合を考える。発電機モデルには，**??**に示される発電機 1 と発電機 3 の定数を用いる。

??と**??**の 2 つの潮流計算結果に対して，発電機 1 と発電機 3 の機械入力，界磁電圧，回転子偏角，内部電圧の定常値を式 (**??**) に基づいて計算する。計算された定常値を**??**と**??**に示す。同様に，式**??**に基づいて負荷のインピーダンス値を逆算すると，**??**と**??**の 1 行目のように求められる。潮流計算の結果により，発電機の内部状態や外部入力の定常値，負荷のインピーダンス値が異なることに注意されたい。

発電機の内部状態の定常値を摂動して初期値に設定しよう。具体的には

$$\begin{bmatrix} \delta_1(0) \\ \delta_3(0) \end{bmatrix} = \begin{bmatrix} \delta_1^\star + \frac{\pi}{6} \\ \delta_3^\star \end{bmatrix}, \qquad \begin{bmatrix} E_1(0) \\ E_3(0) \end{bmatrix} = \begin{bmatrix} E_1^\star + 0.1 \\ E_3^\star \end{bmatrix} \tag{1.38}$$

とする。ただし，角周波数偏差の初期値は 0 とする。このとき，2 つの潮流計算結果に対する角周波数偏差の初期値応答を**??**に示す。青の実線が発電機 1 の角周波数偏差，赤の破線が発電機 3 の角周波数偏差を表す。どちらの定常潮流状態でも同程度の周波数の振動が生じていることがわかる。なお，基準角周波数は 60 [Hz] に設定されているため，角周波数偏差の 0.015 [pu] は，0.9 [Hz] に等しい。

表 1.5 負荷のインピーダンス値 [pu]

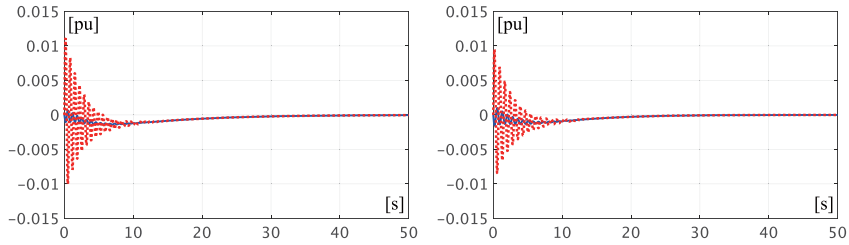
負荷	$\mathsf{Re}[\boldsymbol{z}^\star_{\mathrm{load2}}]$	$\mathsf{Im}[\boldsymbol{z}^\star_{\mathrm{load2}}]$
定常	1.3293	0
増加	1.3426	0
減少	1.3160	0

(a) **??**の潮流計算結果

表 1.6 負荷のインピーダンス値 [pu]

負荷	$\mathsf{Re}[\boldsymbol{z}^\star_{\mathrm{load2}}]$	$\mathsf{Im}[\boldsymbol{z}^\star_{\mathrm{load2}}]$
定常	1.3224	0
増加	1.3356	0
減少	1.3092	0

(a) **??**の潮流計算結果



(a) **??**の定常潮流状態　　(b) **??**の定常潮流状態

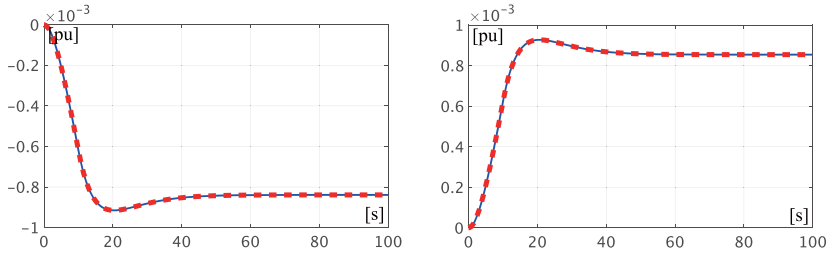
図 1.2　初期値変動に対する角周波数偏差の時間応答
（青実線：$\Delta\omega_1$，赤破線：$\Delta\omega_3$）

1.4.2　負荷モデルのパラメータ変動に関する応答

定常潮流状態にある電力系統モデルに対して，負荷モデルの定数を 1 つでも変化させると，一般にすべての母線における電圧フェーザと電流フェーザの値が変化する。このとき，一般に系統全体での電力の需給がバランスしなくなるため，与えられた界磁電圧の値に応じて，機械入力の値を適切に修正しない限りは，各発電機の角周波数偏差は 0 に収束しない。このことを確認してみよう。

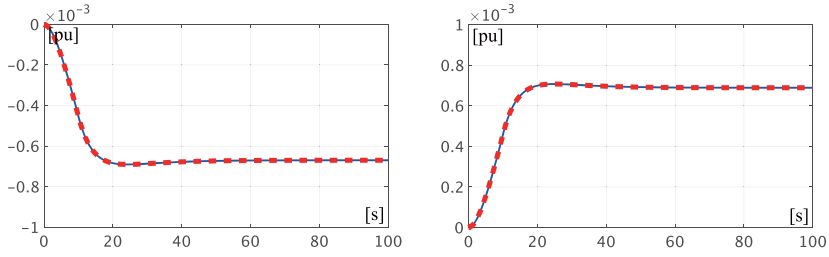
例 1.4　(負荷のインピーダンス変化に対する電力系統モデルの時間応答)
例**??**と同じ設定で負荷のインピーダンス値が変化した場合の角周波数偏差の時間応答を計算してみよう。具体的には，潮流計算の結果から逆算された負荷の抵抗を 1%増加または減少させて時間応答を計算する。増加後と減少後の負荷のインピーダンス値は，**??**と**??**の 2 行目と 3 行目に示されている。

計算結果を**??**と**??**に示す。青の実線は発電機 1 の角周波数偏差，赤の破線は発電機 3 の角周波数偏差を表している。どの場合においても，2 つの発電機の角周波数偏差が同期して変化していることがわかる。また，抵抗が増加すると有効電力の消費が一般に大きくなるため，発電機の周波数は低下する。抵抗が減少する場合は逆である。発電機の機械入力が抵抗が変



(a) 抵抗が 1%増加した場合　　(b) 抵抗が 1%減少した場合

図 1.3 負荷の変化に対する角周波数偏差の時間応答
（??の定常潮流状態，線種は??と同様）



(a) 抵抗が 1%増加した場合　　(b) 抵抗が 1%減少した場合

図 1.4 負荷の変化に対する角周波数偏差の時間応答
（??の定常潮流状態，線種は??と同様）

化する前の値で固定されているため，有効電力の需給バランスが取れなくなり，角周波数偏差の定常値は非零となっていることに注意されたい。なお，負荷の抵抗の変化比率は??と??の両者で等しいが，生じる角周波数偏差の値は異なっている。これは電力系統モデルの平衡点の選び方によって，外乱に対する感度（安定度）が変化することを示唆している。

例??では，例??の結果とは異なり，発電機の角周波数偏差が非零の定常値をもつ。この角周波数偏差を 0 にするためには，発電機群の機械入力もしくは界磁電圧の値を適切に調整する必要がある。これに対して，角周波数偏差の定常

値を 0 にすることは，有効電力の需要と供給をバランスさせることであるため，発電機群の機械入力を調整する制御アルゴリズムにより周波数制御が行われることが一般的である。ただし，すべての負荷の変化量を正確に計測することは現実的に困難である。したがって，積分器に基づく制御動作により，需給バランスを達成する機械入力の値を自動探索するフィードバック制御が必要となる。その詳細は，**??**節や**??**節で後述する。

1.4.3 地絡による応答

（**1**） **地絡とは** 物体と送電線の接触や落雷などをきっかけに電気回路が大地と接触して，大地に大きな電流が流れる現象は**地絡**（ground fault）と呼ばれる。地絡が長時間継続すると接続されている機器や設備などが損傷してしまうため，電力系統では地絡の発生を検知して地絡電流を遮断する装置が高速に作動する。地絡電流を除去した後で遮断を解除すれば，地絡発生前の電力系統運用に復帰する。

地絡の発生を検知して地絡を除去するまでに要する時間は，概ね 70 [ms] 程度である。この間に大地に流れる地絡電流が，電力系統の状態を大きく変動させるシビアな外乱となる。なお，地絡による外乱は外部入力としてモデル化されるものではなく，「地絡が継続している時刻では別の電力系統モデルに切り替わる変動」としてモデル化される。

（**2**） **母線地絡の定式化** 本書で扱う母線に発生する地絡は，地絡が継続している時刻において，その母線の電圧フェーザの値が 0 に拘束されるものとしてモデル化される。以下では，N 個の母線で構成される電力系統モデルに対して，一般性を失うことなく，母線 1 に地絡が発生するものとして説明する。また，地絡は時刻 0 [s] に発生し，時刻 t_0 [s] まで継続するものとする。このとき，式**??**の潮流状態は，$t < 0$ と $t \geqq t_0$ の時刻では，式**??**の平常時の代数方程式を満たすが，地絡が継続している $t \in [0, t_0)$ の時刻では，地絡母線の電流フェーザ $\boldsymbol{I}_1(t)$ に関する方程式を除いた

$$\begin{bmatrix} \boldsymbol{I}_2(t) \\ \vdots \\ \boldsymbol{I}_N(t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{Y}_{22} & \cdots & \boldsymbol{Y}_{2N} \\ \vdots & \ddots & \vdots \\ \boldsymbol{Y}_{N2} & \cdots & \boldsymbol{Y}_{NN} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_2(t) \\ \vdots \\ \boldsymbol{V}_N(t) \end{bmatrix} \tag{1.39a}$$

の代数方程式と，地絡母線の電圧フェーザに対する拘束条件

$$|\boldsymbol{V}_1(t)| = 0 \tag{1.39b}$$

を満たさなければならない。なお，地絡が発生していない母線や時刻では，電流フェーザと電圧フェーザは，各機器のモデルとして表される微分方程式または代数方程式にしたがう。これは平常時と同じである。すなわち

- 地絡が発生していない時刻では，平常の電力系統モデル
- 地絡が継続している時刻では，地絡が発生している母線とその母線に接続されている機器や送電線を取り除いた電力系統モデル

を切り替えて用いることに等しい。

地絡時に機器 1 から母線 1 に流入する電流フェーザ $\boldsymbol{I}_1(t)$ は，機器の動特性から定められる。具体的には，機器が発電機である場合には，$t \in [0, t_0)$ の時刻では，$|\boldsymbol{V}_1(t)|$ を 0 とした

$$\begin{cases} \dot{\delta}_1 = \omega_0 \Delta\omega_1 \\ M_1 \Delta\dot{\omega}_1 = -D_1 \Delta\omega_1 + P_{\mathrm{mech1}} \\ \tau_1 \dot{E}_1 = -\frac{X_1}{X_1'} E_1 + V_{\mathrm{field1}} \end{cases} \tag{1.40}$$

の微分方程式にしたがって発電機の内部状態が時間発展する。このとき，電流フェーザ $\boldsymbol{I}_1(t)$ は，発電機の出力として

$$|\boldsymbol{I}_1(t)| = \frac{E_1(t)}{X_1'}, \qquad \angle \boldsymbol{I}_1(t) = \delta_1(t) - \frac{\pi}{2}$$

で与えられる。同様に，機器 1 が定インピーダンスの負荷モデルである場合には，式**??**の $|\boldsymbol{V}_1(t)|$ が 0 であるため，$|\boldsymbol{I}_1(t)|$ も 0 となる。他の負荷モデルの場合も基本的に同様であるが，定電力の負荷モデルでは $|\boldsymbol{I}_1(t)|$ が無限大となるた

め，数値シミュレーションでは数値的な不安定性に注意する必要がある。なお，母線 1 から大地に流れる地絡電流フェーザは

$$\boldsymbol{I}'_1(t) := \boldsymbol{I}_1(t) - \sum_{j=2}^{N} \boldsymbol{Y}_{1j}\boldsymbol{V}_j(t), \qquad t \in [0, t_0)$$

と表すことができる。

地絡電流フェーザの値は，電力系統モデルの数値シミュレーションを実行する目的では計算する必要はない。一方で，地絡母線に発電機が接続されている場合には，地絡が除去される時刻 t_0 における発電機の内部状態の値を計算することが重要であるため，式**??**の微分方程式は解く必要がある。地絡が発生する時刻の電力系統モデルの内部状態，すなわち，初期時刻における各発電機の内部状態には，任意の潮流状態における値を設定すれば良い。本書では，潮流計算の結果として求められた適当な定常潮流状態の値を設定する。なお，地絡が発生する時刻や除去される時刻では，各発電機の内部状態は連続であるが，各母線の電圧フェーザや電流フェーザは不連続に変化する。

以上の母線地絡に対する時間応答の計算は，つぎの手順にまとめられる。

(a) 潮流計算で求められた定常潮流状態における変数値を各発電機の内部状態の初期値として設定する。

(b) 地絡が継続している $t \in [0, t_0)$ の時刻では，地絡が発生している母線とその母線に接続されている機器や送電線を取り除いた電力系統モデルによって時間発展を計算する。

(c) 地絡が発生している母線に発電機が接続されている場合には，地絡が継続している $t \in [0, t_0)$ の時刻では母線電圧フェーザを 0 に設定して，その発電機の内部状態の時間発展を計算する。

(d) 時刻 t_0 における各発電機の内部状態の値を設定して，すべての機器が接続された平常の電力系統モデルにより地絡除去後の時間発展を計算する。

この手順にしたがって母線地絡による時間応答を計算してみよう。

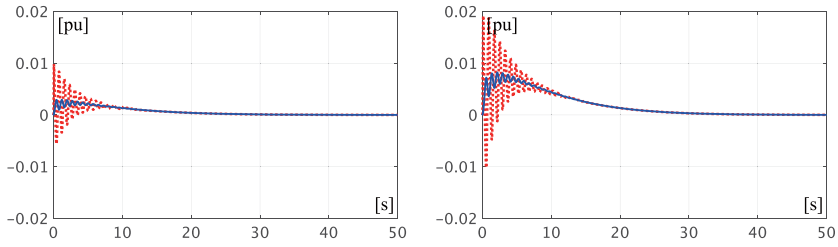
例 1.5 (母線地絡に対する電力系統モデルの時間応答) 例**??**や例**??**と同

じ設定で，母線の地絡に対する角周波数偏差の時間応答を計算してみよう。具体的には，電力系統が潮流計算で求められた 2 つの定常潮流状態を初期値に設定して，母線 1 に地絡が生じた場合の時間応答を計算する。比較のため，地絡が除去されるまでの時間が 50 [ms] である場合と，100 [ms] である場合の 2 通りを考える。

計算結果を**??**と**??**に示す。青の実線は発電機 1 の角周波数偏差，赤の破線は発電機 3 の角周波数偏差を表している。**??**から，**??**の定常潮流状態では，発電機 3 の周波数の振動が大きいことがわかる。特に，地絡の継続時間が 100 [ms] である場合にはより大きな振動が生じている。発電機 3 の振動がより大きい理由は，**??**に示されているように，発電機 1 の慣性定数が 100 [s] と大きいのに対して，発電機 3 の慣性定数は 12 [s] と小さいためである。すなわち，慣性が大きい発電機の振動が慣性が小さい発電機の大きな振動を引き起こしている。

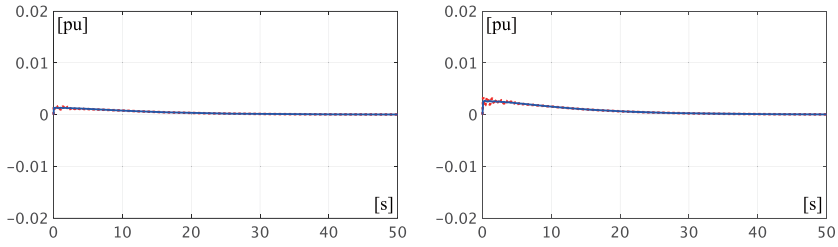
一方で，**??**の定常潮流状態では，同じ地絡に対しても周波数の振動が小さいことが**??**からわかる。この理由は，**??**の定常潮流状態では，小さな慣性をもつ発電機 3 から負荷で消費される有効電力の大半を供給していたのに対して，**??**の定常潮流状態では，大きな慣性をもつ発電機 1 から有効電力の大半を供給していたためである。一般に，同期発電機は，供給する電力が最大に近づくほど外乱に対する感度も高くなる特性をもつ。**??**の定常潮流状態では，慣性の小さい発電機 3 の安定度が相対的に高いため，地絡に対する角周波数偏差の感度が低くなっている。

例**??**から，定常的な潮流状態（平衡点）に応じて，地絡に対する電力系統モデルの安定度が変化することがわかる。特に，例**??**では，**??**の定常潮流状態が送電損失の観点で優れていたのに対して，例**??**では，**??**の定常潮流状態が地絡に対する系統安定度の観点で優れていることがわかる。これらの例では，経済性や安定度などに関するトレードオフを考慮して，望ましい平衡点を探索することの重要性が示唆されている。なお，地絡は特定の母線や送電線のみに発生



(a) 50 [ms] の地絡　　(b) 100 [ms] の地絡

図 1.5　地絡に対する角周波数偏差の時間応答
（**??**の定常潮流状態，線種は**??**と同様）



(a) 50 [ms] の地絡　　(b) 100 [ms] の地絡

図 1.6　地絡に対する角周波数偏差の時間応答
（**??**の定常潮流状態，線種は**??**と同様）

するものではなく，様々な地点で発生する可能性があるため，電力系統全体の安定度を適切な意味で向上させる必要がある。そのための制御機構は，**??**節や**??**節で説明する。

なお，例**??**から例**??**のシミュレーション結果から，内部状態が発散することなくある定常状態に落ち着いた場合には，「すべての発電機の角周波数偏差が同じ値に収束していること」がわかる。同様の結果は，例**??**でも観察されている。定常潮流状態におけるこの周波数の同期は，電力系統モデルにおいて普遍的な現象である。**??**節や**??**節で後述する周波数制御では，定常潮流状態で周波数が自動的に同期することを前提に制御アルゴリズムが構築される。

1.5 定常的な潮流状態における母線電圧の同期 ⋆

本節では，**??**節で観察された周波数の同期現象を送電網のグラフ構造の観点から数学的に考察する。定理**??**の結果に基づき，つぎの定義を導入する。

定義 1.1 (定常潮流状態と母線電圧の同期) 式**??**の連立方程式によって機器群が結合された電力系統モデルを考える。すべての母線 i に対して

$$\frac{dP_i}{dt}(t)=0,\quad \frac{dQ_i}{dt}(t)=0,\quad \frac{d|\boldsymbol{V}_i|}{dt}(t)=0,\quad \frac{d^2\angle\boldsymbol{V}_i}{dt^2}(t)=0 \quad (1.41)$$

がすべての $t\geqq 0$ に対して成り立つとき，電力系統は**定常潮流状態**にあると呼ぶ [†2]。また，電力系統が定常潮流状態にあり，かつ

$$\frac{d\angle\boldsymbol{V}_i}{dt}(t)=\frac{d\angle\boldsymbol{V}_j}{dt}(t) \qquad (1.42)$$

が成り立つとき，定常潮流状態で母線 i と母線 j は**同期する**と呼ぶ。

定理**??**に示されているように，発電機母線に対しては，式**??**が成り立つことと，発電機の内部状態と外部入力が定常状態にあることは等価である。また，任意に選ばれた母線の組 (i,j) が定義**??**の意味で同期しているのであれば，すべての発電機の角周波数偏差が同じ値に収束することが結論づけられる。なお，いずれの母線に対しても，式**??**が成り立つことは，電流フェーザ $\boldsymbol{I}_i$ に対して

$$\frac{d|\boldsymbol{I}_i|}{dt}(t)=0,\qquad \frac{d^2\angle\boldsymbol{I}_i}{dt^2}(t)=0,\qquad \frac{d\angle\boldsymbol{I}_i}{dt}(t)=\frac{d\angle\boldsymbol{V}_i}{dt}(t)$$

が成り立つことを意味する。このことは

$$|P_i(t)+\boldsymbol{j}Q_i(t)|=|\boldsymbol{V}_i(t)||\boldsymbol{I}_i(t)|,\qquad \angle(P_i(t)+\boldsymbol{j}Q_i(t))=\angle\boldsymbol{V}_i(t)-\angle\boldsymbol{I}_i(t)$$

[†2] この「定常潮流状態」の数学的な定義は本書独自のものであり，電力系統工学で一般に用いられる定義ではないことに注意されたい。

であることから簡単に確認することができる。

母線 i と送電線で結ばれている隣接母線の集合を $\mathcal{N}_i$ と表す。すなわち

$$\boldsymbol{Y}_{ij} = 0, \qquad \forall j \notin \mathcal{N}_i$$

であるものとする。母線 i と送電線で結ばれている隣接母線の数は $|\mathcal{N}_i|$ であり，このことを「母線 i の**次数**（degree）は $|\mathcal{N}_i|$ である」と呼ぶ。また，電力系統モデルは定常潮流状態にあることを仮定して

$$\angle \boldsymbol{V}_i(t) = \Omega_i t + \phi_i$$

と表す。ただし，Ω_i と ϕ_i は定数である。このとき，式**??**の母線 i に関する電力バランス方程式は

$$P_i + \boldsymbol{j} Q_i = \sum_{j=1}^{N} \overline{\boldsymbol{Y}}_{ij} |\boldsymbol{V}_i| |\boldsymbol{V}_i| e^{\boldsymbol{j}\{(\Omega_i - \Omega_j)t + \phi_i - \phi_j\}}$$

となる。ここで，定常潮流状態を仮定する場合には，母線 i に供給される有効電力と無効電力，母線電圧フェーザの絶対値はすべて定数であり，それらを $P_i^\star$，$Q_i^\star$，$|\boldsymbol{V}_i^\star|$ と表せば，この電力バランス方程式は

$$\underbrace{\frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} r_{ij} e^{\boldsymbol{j}(\Omega_{ij} t + \Phi_{ij})}}_{\boldsymbol{C}_i(t)} = \boldsymbol{z}_i \tag{1.43}$$

と変形できる。ただし，母線 i と j の電圧フェーザの周波数差を

$$\Omega_{ij} := \Omega_i - \Omega_j$$

と表している。また，r_{ij}，Φ_{ij}，$\boldsymbol{z}_i$ はすべて定数であり

$$r_{ij} := |\boldsymbol{V}_i^\star| |\boldsymbol{V}_j^\star| |\boldsymbol{Y}_{ij}|, \qquad \Phi_{ij} := \phi_i - \phi_j - \angle \boldsymbol{Y}_{ij},$$
$$\boldsymbol{z}_i := \frac{\left\{ P_i^\star - \mathsf{Re}[\boldsymbol{Y}_{ii}] |\boldsymbol{V}_i^\star|^2 + \boldsymbol{j}\left(Q_i^\star + \mathsf{Im}[\boldsymbol{Y}_{ii}] |\boldsymbol{V}_i^\star|^2 \right) \right\}}{|\mathcal{N}_i|}$$

で定義される。

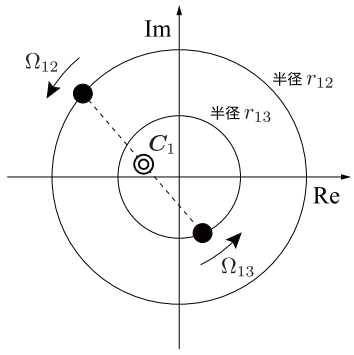


図 1.7 母線 1 が母線 2 と母線 3 に接続している場合

以下では，式**??**の方程式から，隣接する母線との同期を表す等式として，すべての $j \in \mathcal{N}_i$ に対して Ω_{ij} が 0 であること，すなわち

$$\Omega_i = \Omega_j, \qquad \forall j \in \mathcal{N}_i \tag{1.44}$$

を導くことを考える。ここで，式**??**は「原点を中心とする半径 r_{ij} の円周上を，初期位相 Φ_{ij}，角速度 Ω_{ij} で等速運動する $|\mathcal{N}_i|$ 個の点の重心 $\boldsymbol{C}_i(t)$ が，複素平面上のある点 $\boldsymbol{z}_i$ で不変であること」を表している。**??**には，母線 1 が母線 2 と母線 3 に接続している場合の関係を例示している。この事実に注目するとつぎの結果が導ける。

補題 1.2 (電力潮流方程式から導かれる母線の同期) 実定数 $r_{ij}, \Omega_i, \Omega_j, \Phi_{ij}$ に対して，式**??**の $\boldsymbol{C}_i(t)$ を考える。ただし，$r_{ij} > 0$ とする。このとき，$|\mathcal{N}_i| = 1$ であるならば，$\boldsymbol{C}_i(t)$ が t に依らない定数であることは，式**??**と等価である。また，$|\mathcal{N}_i| = 2$ であるならば，$\boldsymbol{C}_i(t)$ が t に依らない定数であることは，式**??** が成り立つこと，または

$$\Omega_{j_1} = \Omega_{j_2}, \qquad r_{ij_1} = r_{ij_2}, \qquad |\Phi_{ij_1} - \Phi_{ij_2}| = \pi \tag{1.45}$$

と等価である。ただし，$\mathcal{N}_i = \{j_1, j_2\}$ である。さらに，$|\mathcal{N}_i| = 3$ であるならば，$\boldsymbol{C}_i(t)$ が t に依らない定数であることは，式**??**が成り立つこと，ま

たは，$\Omega_i = \Omega_{j_3}$ を満たす $j_3 \in \mathcal{N}_i$ に対して，式**??**が成り立つことと等価である。ただし，$\mathcal{N}_i \setminus \{j_3\} = \{j_1, j_2\}$ である。

証明 脚注の補題**??** †3 を適用することで，$|\mathcal{N}_i| = 1$ と $|\mathcal{N}_i| = 2$ の場合の事実を示すことができる。したがって，以下では $|\mathcal{N}_i| = 3$ の場合について考える。表記の簡単化のため，$j \in \{1,2,3\}$ とし，r_{ij}，Φ_{ij}，Ω_i，$\boldsymbol{C}_i$ をそれぞれ r_j，Φ_j，Ω_0，$\boldsymbol{C}_0$ と表す。まず

$$\Omega_j \neq \Omega_0, \qquad \forall j \in \{1,2,3\} \tag{1.46}$$

であるとき，式**??**を満たす Ω_j は存在しないことを示す。補題**??**を適用すると，式**??**が成り立つとき，式**??**は

$$\Omega_1 = \Omega_2 = \Omega_3, \qquad \sum_{j=1}^{3} r_j e^{\boldsymbol{j}\Phi_j} = 0$$

と等価であることがわかる。しかしながら，$\Omega_1 = \Omega_2 = \Omega_3$ である場合には，$|\mathcal{N}_i| = 1$ のときと同様にして，$\boldsymbol{C}_i(t)$ が t に依らない定数であることと式**??**が等価であることが導かれる。これは式**??**に矛盾する。

以上から，**??**の否定として，ある $j \in \{1,2,3\}$ に対して $\Omega_0 = \Omega_j$ である場合のみを考えれば良い。特に，j に関する対称性から，一般性を失うことなく，$\Omega_0 = \Omega_3$ である場合を考える。このとき

---

†3 証明は付録を参照されたい。

補題 1.3 実定数 r_i，ω_i，ϕ_i に対して

$$\boldsymbol{C}_n(t) := \sum_{i=1}^{n} r_i e^{\boldsymbol{j}(\omega_i t + \phi_i)}$$

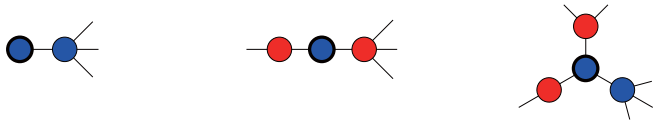
とする。ただし，$r_i > 0$ および $\phi_i \in [0, 2\pi)$ とする。このとき，$\boldsymbol{C}_1$ が t に依らない定数であるための必要十分条件は $\omega_1 = 0$ である。また，$\boldsymbol{C}_2$ が t に依らない定数であるための必要十分条件は $\omega_1 = \omega_2 = 0$ または

$$\omega_1 = \omega_2, \qquad r_1 = r_2, \qquad |\phi_2 - \phi_1| = \pi$$

である。さらに，ω_1，ω_2，ω_3 がすべて 0 でないとき，$\boldsymbol{C}_3$ が t に依らない定数であるための必要十分条件は

$$\omega_1 = \omega_2 = \omega_3, \qquad \sum_{i=1}^{3} r_i e^{\boldsymbol{j}\phi_i} = 0$$

である。



(a) 次数 1 の母線 (b) 次数 2 の母線 (c) 次数 3 の母線

図 1.8 母線の次数に応じた隣接母線との同期

$$\boldsymbol{C}_0(t) = \frac{1}{3}\left\{ r_3 e^{\boldsymbol{j}\Phi_3} + \sum_{j=1}^{2} r_j e^{\boldsymbol{j}\{(\Omega_0-\Omega_j)t+\Phi_j\}} \right\}$$

であることから，t に関する不変性は，$|\mathcal{N}_i| = 2$ の場合と同様に議論できる。したがって，$\boldsymbol{C}_0(t)$ が t に依らない定数であることは，式**??**または

$$\Omega_1 = \Omega_2, \qquad r_1 = r_2, \qquad |\Phi_1 - \Phi_2| = \pi$$

と等価である。以上より題意が示される。 □

補題**??**は，注目する母線の次数が 1 のとき，すなわち，**??**(a) のような端点の母線については，その母線と隣の母線が同期することを示している。また，注目する母線（太線で示されたノード）の次数が 2 のとき，すなわち，**??**(b) のような鎖状経路にある母線については，少なくともその両隣の母線が同期する。さらに，注目する母線の次数が 3 のとき，すなわち，**??**(c) のような 3 本の送電線で結ばれている節の母線については，隣接する母線のうち少なくとも 1 つが注目する母線と同期する。これは，任意に選ばれた 3 つの母線のみが同期し，残りの 1 つの母線は同期しないような状況や，どの母線の組も同期しないような状況は生じないことも意味している。

注目する母線の次数が 4 以上である場合にも同様の解析を行うことは可能である。しかしながら，得られる条件が Ω_i や Ω_j に関する高次の方程式になることや，任意に選ばれた一部の隣接母線のみが同期する場合などの複数の組み合わせが存在するため，同期に関する等価条件を書き下すことは一般に煩雑となる。ただし，注目する母線 i とそれに隣接する $|\mathcal{N}_i|$ 個の母線に対して，いずれ

(a) (b)

図 1.9 木構造をもつ送電網における母線の同期

かの $|\mathcal{N}_i| - 1$ 個の隣接母線が母線 i と同期するのであれば，残り 1 個の隣接母線も同期することは一般に示される。

補題**??**で示されている次数が 3 以下の条件を組み合わせて用いることによって，次数が 4 以上の母線が送電網に含まれている場合にも，すべての母線の同期が示される場合は存在する。例えば，つぎの事実を示すことができる。

定理 1.4 (木構造の送電網における母線の同期) 式**??**の連立方程式によって機器群が結合された電力系統モデルを考える。送電網のグラフが木構造をもつとき [†4]，定常潮流状態においてすべての母線は同期する。

証明 **??**(a) の太線で示された端点の母線に注目する。母線の次数は 1 であるから，その隣の母線は端点の母線と同期する。つぎに，端点の隣の母線が鎖状経路にある場合には，その母線の次数は 2 であるため，少なくともその両隣の母線は同期する。これを繰り返していくことで，**??**(a) のように，端点と次数が 3 以上の節の母線を結ぶ鎖状経路において，すべての母線の同期が示される。

同様に，別の端点から次数が 3 以上の節に存在するすべての母線は同期するため，**??**のように，太線で示される節の母線に連結するすべての鎖状経路の母線はすべて同期することがわかる。この議論を繰り返せば，木を構成するすべての母線が同期することが示される。 □

定理**??**に示されている木構造をもつ送電網の母線には次数の制限はない。このように，仮に次数が 4 以上の母線が送電網に含まれていたとしても，グラフ

[†4] グラフ理論において，連結であり閉路をもたないグラフは**木**（tree）と呼ばれる。

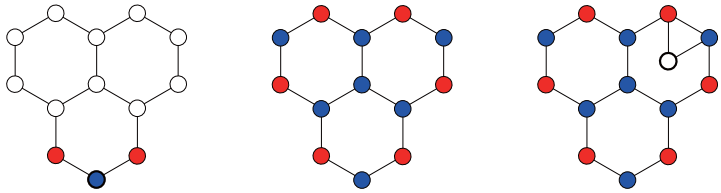


図 1.10 ハニカム構造をもつ送電網における母線の同期

構造の情報だけを用いて，すべての母線の同期を演繹できる場合がある。同様に，つぎの事実を示すことができる。

定理 1.5 (円環構造の送電網における母線の同期) 式**??**の連立方程式によって機器群が結合された電力系統モデルを考える。送電網のグラフが円環構造をもち，かつ，母線の総数が奇数であるとき，定常潮流状態においてすべての母線は同期する。

証明 ある母線に注目するとその両端の母線の同期が示される。これを繰り返すことにより，母線の総数が奇数である場合にすべての母線の同期が示される。 □

定理**??**は，円環構造の送電網に対して，母線の総数が奇数である場合は，各送電網のアドミタンスの値などに依らず，すべての母線が同期することを示している。一方で，母線の総数が偶数である場合には，送電線のアドミタンス値などの追加の情報を用いない限りは，円環構造の送電網に対してもすべての母線の同期を結論づけることはできない。後述する例**??**では，母線の総数が偶数である場合にも，アドミタンスの値などの一部の情報が与えられるだけで，すべての母線の同期が示され得ることを示す。次数 3 の母線を含む送電網に補題**??**を適用した例としてつぎを示す。

例 1.6 (ハニカム構造の送電網における母線の同期) **??**(a) に示される

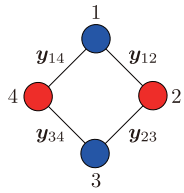


図 1.11 円環構造をもつ送電網における 4 母線の同期

ハニカム構造をもつ送電網に対して，定常潮流状態における母線の同期を考えよう。最下部の母線に注目すると，その両隣の母線の同期がわかる。これらの同期する母線を赤で示し，最下部の母線と同期するものを青で示す。このとき，補題**??**を各母線に適用していくことによって，同期するすべての母線群を**??**(b) のように色分けすることができる。しかしながら，この場合には，グラフ構造の情報だけで赤の母線群と青の母線群の同期を結論づけることはできない。一方で，**??**(c) のように母線が 1 つ追加された送電網の場合には，その追加された母線に注目することで，両隣の赤と青の母線の同期が導かれる。したがって，**??**(c) のグラフ構造の場合には，すべての母線の同期が示される。

例**??**において，一部のグラフ構造の差異により同期する母線の対称性が崩れることで，電力系統全体での母線の同期が演繹されている点は興味深い。つぎの例では，アドミタンスの値まで考慮した「定量的な送電網の対称性」の観点から母線の同期を考察する。

例 1.7 (円環構造の送電網における 4 つの母線の同期) **??**に示される円環構造をもつ送電網に対して，定常潮流状態における母線の同期を考えよう。母線の数は偶数であるため，定理**??**のようにグラフ構造の情報だけですべての母線の同期を示すことはできない。ただし，**??**の赤と青で示されているように，少なくとも互い違いの母線がそれぞれ同期することは補題**??**からわかる。したがって，すべての母線の同期を示すためには，少なく

とも 1 つの母線に対して式**??**のいずれかの条件が満たされないことを示せば良い。

母線 i と母線 j を結ぶ送電線のアドミタンスを $\boldsymbol{y}_{ij}$ と表す。このとき，各母線に対する式**??**中央の条件は

$$|\boldsymbol{V}_2^\star||\boldsymbol{y}_{12}| = |\boldsymbol{V}_4^\star||\boldsymbol{y}_{14}|, \qquad |\boldsymbol{V}_1^\star||\boldsymbol{y}_{12}| = |\boldsymbol{V}_3^\star||\boldsymbol{y}_{23}|,$$
$$|\boldsymbol{V}_2^\star||\boldsymbol{y}_{23}| = |\boldsymbol{V}_4^\star||\boldsymbol{y}_{34}|, \qquad |\boldsymbol{V}_3^\star||\boldsymbol{y}_{34}| = |\boldsymbol{V}_1^\star||\boldsymbol{y}_{14}|$$

と書き下すことができる。これを行列形式で表現すれば

$$\underbrace{\begin{bmatrix} 0 & |\boldsymbol{y}_{12}| & 0 & -|\boldsymbol{y}_{14}| \\ -|\boldsymbol{y}_{12}| & 0 & |\boldsymbol{y}_{23}| & 0 \\ 0 & -|\boldsymbol{y}_{23}| & 0 & |\boldsymbol{y}_{34}| \\ |\boldsymbol{y}_{14}| & 0 & -|\boldsymbol{y}_{34}| & 0 \end{bmatrix}}_{S} \begin{bmatrix} |\boldsymbol{V}_1^\star| \\ |\boldsymbol{V}_2^\star| \\ |\boldsymbol{V}_3^\star| \\ |\boldsymbol{V}_4^\star| \end{bmatrix} = 0$$

となる。この方程式を満たす正のベクトル $(|\boldsymbol{V}_1^\star|, \ldots, |\boldsymbol{V}_4^\star|)$ が存在するための必要条件は，左辺の S が正則でないことである。ここで，列ベクトルの疎構造から，S が非正則であるためには

$$|\boldsymbol{y}_{12}||\boldsymbol{y}_{34}| = |\boldsymbol{y}_{14}||\boldsymbol{y}_{23}| \tag{1.47}$$

でなければならない。したがって，アドミタンス行列がこの条件を満たさない限り，すべての母線は同期する。なお，式**??**が満たされる場合には，所望の電圧フェーザの絶対値を定めることができるため，式**??**中央の条件を満たす $(|\boldsymbol{V}_1^\star|, \ldots, |\boldsymbol{V}_4^\star|)$ が存在するための必要十分条件が，式**??**であることも示される。

つぎに，式**??**右の条件は，母線 1 と母線 3 に注目すれば

$$|\phi_2 - \phi_4 + \angle\boldsymbol{y}_{12} - \angle\boldsymbol{y}_{14}| = \pi, \qquad |\phi_2 - \phi_4 + \angle\boldsymbol{y}_{23} - \angle\boldsymbol{y}_{34}| = \pi$$

と書き下すことができる。同様に，母線 2 と母線 4 に注目すれば

$$|\phi_1 - \phi_3 + \angle \boldsymbol{y}_{12} - \angle \boldsymbol{y}_{23}| = \pi, \qquad |\phi_1 - \phi_3 + \angle \boldsymbol{y}_{14} - \angle \boldsymbol{y}_{34}| = \pi$$

が得られる。一般に，対地静電容量が十分に小さいとき，アドミタンスの実部であるコンダクタンス成分は非負であり，虚部であるサセプタンス成分は負であること，すなわち

$$\angle \boldsymbol{y}_{ij} \in \left[-\frac{\pi}{2}, 0\right)$$

であることに注意すると，以上の条件を満たす $(\phi_1, \ldots, \phi_4)$ が存在するための必要十分条件が

$$\angle \boldsymbol{y}_{12} - \angle \boldsymbol{y}_{14} = \angle \boldsymbol{y}_{23} - \angle \boldsymbol{y}_{34} \tag{1.48}$$

であることが導ける。したがって，アドミタンス行列がこの条件を満たさない限り，すべての母線は同期する。

以上の議論から，定常潮流状態において同期しない母線の組が 1 つ以上存在するための必要十分条件は，式**??**かつ式**??**であることがわかる。この 2 つの条件は，**??**の送電網がアドミタンスの値に関する特異的な対称性をもつときにのみ，互い違いの母線のみが同期するような状況が起こることを示唆している。

例**??**から，式**??**の条件は送電網のアドミタンスの値に関する特異的な対称性を表すことがわかる。実応用においては，各母線の次数が高くない疎な送電網に対してすべての母線が定常潮流状態で同期することは，そのグラフ構造に特異的な対称性が存在しない限りは普遍的な事実である。実際，著者らが知る限りにおいて，現実的な値に設定されたいかなる電力系統モデルのパラメータに対しても，定常潮流状態におけるすべての母線の同期が数値的に確かめられている。

1.6 潮流計算の実装法

電力システムの解析や制御に関するシミュレーション環境を構築することを考える。複雑な電力システムのシミュレーションを間違いなく実施するためには，機能ごとに分割されたモジュール群として記述するオブジェクト指向型の思考法やプログラミング技法が役に立つ。本節では，このような考え方に基づいた潮流計算の実装法について述べる。

1.6.1 代数方程式の解き方

潮流計算のためには，式**??**の代数方程式を解く必要がある。本項ではまず簡単な例題を用いて，代数方程式の解を MATLAB で探索する方法を示す。

例 1.8 (代数方程式の解の探索)　代数方程式

$$x^2 - y = 0 \tag{1.49a}$$

$$x^2 + y^2 - 2 = 0 \tag{1.49b}$$

を満たす (x, y) の組を数値計算によって求めることを考える。解は $(-1, 1)$, $(1, 1)$ の 2 つであることに注意しよう。MATLAB で代数方程式を解くために便利なコマンドとして，`optimization toolbox` の `fsolve` がある。このコマンドを用いるためには，式 (**??**) の代数方程式を

$$f(x, y) = 0$$

の形式として表したときの関数 $f(x, y)$ を実装する必要がある。関数 $f(x, y)$ を実装したものがプログラム**??**である。

プログラム 1-1 (func_ex1.m)

```
1  function out = func_ex1(x_in)
2
```

```
3  x = x_in(1);
4  y = x_in(2);
5
6  out = zeros(2, 1);
7  out(1) = x^2 - y;
8  out(2) = x^2 + y^2 - 2;
9
10 end
```

つぎに，この関数を用いて fsolve を実行し，代数方程式の解を探索するプログラムを記述するとプログラム**??**となる。

プログラム 1-2 (main_ex1.m)

```
1  options = optimoptions('fsolve', 'Display', 'iter');
2  x0 = [0.1; 0.5];
3  x_sol = fsolve(@func_ex1, x0, options)
```

ここで，1 行目の options は代数方程式を解く fsolve に与えるオプションを設定しており，この例では最適化の過程を表示させている。また，2 行目の x0 = [0.1; 0.5]; は解の数値的な探索を行う際の初期値を表している。このプログラムを実行するとつぎのような結果が得られる。

実行結果 1.1

```
  Iteration  Func-count     f(x)      （省略）
      0          3          3.2677
      1          6        0.282554
（省略）
      4         15     3.60447e-14

  方程式が解かれました。
（省略）
  x_sol =
      1.0000
      1.0000
```

この実行結果から，反復を行うにつれて $f(x)$ の値が小さくなり，最終的には $f(x)$ がほぼ 0 となる解が見つかっていることがわかる。ここで探索された解は $(1, 1)$ である。この値は実際に代数方程式の解であるが，fsolve

を用いた数値解法では，すべての解が得られるわけではなく，解のうちのいずれか 1 つが得られるに過ぎないことに注意が必要である。

つづいて，もう一方の解が得られる場合を示そう。プログラム**??**の 2 行目を `x0 = [-0.1; 0.5];` と変更して実行すると，つぎのような結果が得られる。

実行結果 1.2

```
 Iteration  Func-count     f(x)
     0          3          3.2677
     1          6        0.282554
（省略）
     4         15     3.60447e-14

方程式が解かれました。
（省略）
x_sol =

   -1.0000
    1.0000
```

この実行結果では解 $(-1, 1)$ が得られている。このように，非線形の代数方程式の数値解法では，初期値によって異なる解が得られる点に注意が必要である。参考として，プログラム**??**の 2 行目で設定されている初期値を `x0 = [-0.5; -0.5];` とした場合の実行結果を見てみよう。

実行結果 1.3

```
  Iteration  Func-count     f(x)      （省略）
     0          3          2.8125
     1          6         1.89068
（省略）
    29         62            1.75

解が見つかりませんでした。
（省略）
x_sol =

    0.0000
   -1.2247
```

この例からわかるように，初期値の与え方によって，解が存在する方程式であっても，正しく解が得られないこともある。実用上は所望の解に近

い初期値を与えることが重要である。

1.6.2 潮流計算のシンプルな実装法

つぎに，潮流計算を行うプログラムの単純な実装法について紹介する。

例 1.9 (潮流計算の実装法) 例**??**の 3 母線で構成される電力系統モデルの潮流計算を行うことを考える。潮流計算も代数方程式を解く計算の一種であるため，例**??**と同様に，$f(x) = 0$ の $f(x)$ の部分を実装することが必要である。

2 つの送電線のアドミタンス値を式**??**の値に設定し，**??**(a) のデータシートに整合する定常潮流状態

$$\left(|\boldsymbol{I}_1^\star|, \angle\boldsymbol{I}_1^\star, |\boldsymbol{V}_1^\star|, \angle\boldsymbol{V}_1^\star, \ldots, |\boldsymbol{I}_3^\star|, \angle\boldsymbol{I}_3^\star, |\boldsymbol{V}_3^\star|, \angle\boldsymbol{V}_3^\star\right)$$

を求めることを考える。アドミタンス行列 $\boldsymbol{Y}$ が与えられるとき，電流は電圧に依存して一意に定まるため，潮流計算は実質的には電圧の組を定める手続きとなる。このことから，代数方程式における変数 x は，定常状態における母線群の電圧フェーザに取ればよいことがわかる。ここでは，すべての母線の電圧フェーザの実部と虚部を並べて x と表すことにする [†5]。このとき，各母線の電圧フェーザや電流フェーザ，有効電力や無効電力はすべて x の関数となる。それらを $\hat{\boldsymbol{V}}_i(x)$, $\hat{\boldsymbol{I}}_i(x)$, $\hat{P}_i(x)$, $\hat{Q}_i(x)$ と表すと，解くべき代数方程式は

[†5] 電圧フェーザの絶対値と偏角の組を x として実装することもできる。

$$|\boldsymbol{V}_1^\star| - |\hat{\boldsymbol{V}}_1(x)| = 0$$

$$\angle\boldsymbol{V}_1^\star - \angle\hat{\boldsymbol{V}}_1(x) = 0$$

$$P_2^\star - \hat{P}_2(x) = 0$$

$$Q_2^\star - \hat{Q}_2(x) = 0$$

$$P_3^\star - \hat{P}_3(x) = 0$$

$$|\boldsymbol{V}_3^\star| - |\hat{\boldsymbol{V}}_3(x)| = 0$$

である。この左辺の関数を実装すると，プログラム**??**のようになる [†6]。

プログラム 1-3 (func_ex2.m)

```
1  function [out, Vhat, Ihat, Phat, Qhat] = func_ex2(x)
2  % x: [Real(V1), Imag(V1),
3  %     Real(V2), Imag(V2),
4  %     Real(V3), Imag(V3)]';
5
6  y12 = 1.3652 - 11.6040j;
7  y23 = -10.5107j;
8  Y = [y12, -y12, 0;
9      -y12, y12+y23, -y23;
10     0, -y23, y23];
11
12 V1abs = 2;
13 V1angle = 0;
14
15 P2 = -3;
16 Q2 = 0;
17
18 P3 = 0.5;
19 V3abs = 2;
20
21 V1hat = x(1) + 1j*x(2);
22 V2hat = x(3) + 1j*x(4);
23 V3hat = x(5) + 1j*x(6);
24
25 Vhat = [V1hat; V2hat; V3hat];
26
27 Ihat = Y*Vhat;
```

[†6] この関数は 5 つの出力引数を持つが，`fsolve` では 1 つ目の出力引数のみが用いられる。残りの引数は結果の確認のために追加したものである。

```
28  PQhat = Vhat.*conj(Ihat);
29  Phat = real(PQhat);
30  Qhat = imag(PQhat);
31
32  out = [V1abs-abs(V1hat); V1angle-angle(V1hat);
33      P2-Phat(2); Q2-Qhat(2);
34      P3-Phat(3); V3abs-abs(V3hat)];
35  end
```

例**??**で述べたように，代数方程式の解を数値的に探索する場合には，初期値の設定が重要である。潮流計算でよく用いられる初期値として**フラットスタート**がある。フラットスタートでは，すべての母線に対して

$$|\hat{\boldsymbol{V}}_i(x)| = 1, \qquad \angle \boldsymbol{V}_i(x) = 0$$

として x の初期値を設定する。これは $\boldsymbol{V}_i$ の実部と虚部をそれぞれ 1 と 0 と設定することに対応する。この初期値を用いて潮流計算の代数方程式を解くプログラムはつぎのようになる。

プログラム 1-4 (main_ex2.m)

```
1  x0 = [1; 0; 1; 0; 1; 0];
2  options = optimoptions('fsolve', 'Display', 'iter');
3  x_sol = fsolve(@func_ex2, x0, options);
4
5  [~, V, I, P, Q] = func_ex2(x_sol);
6  Vabs = abs(V);
7  Vangle = angle(V);
8  display('Vabs:'), display(Vabs')
9  display('Vangle:'), display(Vangle')
10 display('P:'), display(P')
11 display('Q:'), display(Q')
```

プログラム**??**の 1 行目でフラットスタートの初期値を設定している。また，5 行目では代数方程式の解から対応する電圧フェーザ，電流フェーザと有効電力，無効電力の計算を行っている。そして，6 行目以下で電圧フェーザの絶対値，偏角，有効電力，無効電力を表示する。プログラム**??**を実行した結果を示す。

実行結果 1.4

```
 Iteration  Func-count     f(x)      （略）
     0          7           11.25
     1         14          3.28327
（省略）
     5         42      4.19428e-28

方程式が解かれました。
（省略）
Vabs:
    2.0000    1.9918    2.0000

Vangle:
    0.0000   -0.0538   -0.0419

P:
    2.5158   -3.0000    0.5000

Q:
   -0.0347    0.0000    0.1759
```

なお，この実行結果は，**??**(b) に示されている値と一致している。

1.6.3 分割されたモジュール群を用いた潮流計算の実装法

前項では潮流計算のシンプルな実装法を述べたが，この実装では，アドミタンス行列が変更されるたびにプログラム**??**に相当する関数を書き換える必要がある。また，母線の数や種類を変えたい場合にもプログラム全体を書き換えることが必要になる。本項では，大規模で複雑な電力系統に対する潮流計算を行ったり，複数人で分担してプログラムの実装を行ったりできるように，プログラムをモジュール群に分割することを考える。これによりプログラムが構造化され，見通しのよい実装になる。

まず，プログラム**??**のアドミタンス行列の実装を分離することを考えよう。

例 1.10 (アドミタンス行列の実装の分離)

プログラム**??**においてアドミタンス行列を指定している部分を分離しよ

う。このためには変数 Y を関数の入力引数とすればよい。例えば，プログラム**??**のような修正が考えられる。

プログラム 1-5 (func_ex3.m)

```
1  function [out, Vhat, Ihat, Phat, Qhat] = func_ex3(x, Y)
2  （プログラム 3-3 の 12 行目から 34 行目と同じ）
3  end
```

しかしながら，fsolve は最適化するパラメータを唯一の引数とする関数を要求する仕様となっているため，プログラム**??**を直接用いることはできない。この問題を解決するために，プログラム**??**をつぎのように修正する。

プログラム 1-6 (main_ex3.m)

```
1  x0 = [1; 0; 1; 0; 1; 0];
2  options = optimoptions('fsolve', 'Display', 'iter');
3
4  y12 = 1.3652 - 11.6040j;
5  y23 = -10.5107j;
6  Y = [y12, -y12, 0;
7       -y12, y12+y23, -y23;
8       0, -y23, y23];
9
10 func_curried = @(x) func_ex3(x, Y);
11
12 x_sol = fsolve(func_curried, x0, options);
13 [~, V, I, P, Q] = func_curried(x_sol);
14 （プログラム 3-4 の 6 行目以下と同じ）
```

プログラム**??**の 8 行目において用いられている「@(x) x の式」は「引数 x をとり，x を戻り値とする関数」を作成する。このように，関数名を記述せずに生成される関数を**無名関数**という。このとき，無名関数の引数に含まれない変数 Y はワークスペースに存在する定数が常に用いられる。このことにより，関数 func_ex3(x, Y) の 2 つの入力引数 x，Y のうち，1 つ目の引数 x のみを残した新たな関数を生成することができている。このように，複数の引数のうち，いくつかの引数に与える変数を固定し，残った引数のみを引数とする新たな関数を作成することを**カリー化**という。この

ような技術を用いると，任意の数の引数をもつ関数を要求された数の引数をもつ関数として利用することができるようになる。これにより，グローバル変数を用いることなく機能の分離を行うことができる。

つぎに，それぞれの母線における制約条件を計算する関数であるプログラム**??**について考える。プログラム**??**では，各母線の制約条件が直接的に書き下されている。このことから，母線の数を変更するたびにプログラムを書き換える必要がある。さらに，母線の種類による場合分けが暗に含まれており，母線の種類を増やすと場合分けの処理を修正する必要があるという問題点もある。

場合分けの記述を避けながら見透し良くプログラムを記述するためには，オブジェクト指向における多態性の考え方を用いるとよい。つぎの例では実装例を通して多態性の考え方を紹介する。

例 1.11 (多態性を用いた潮流計算の実装例) 潮流計算を行う際，母線の種類の違いによって異なる制約条件が設定される。このとき，各種の母線は「制約条件をもつ」という点では共通である。この共通の性質を，共通の名前をもつメソッドとして実装すると，取り回しのしやすいプログラムを作成することができる。このことを理解するために，スラック母線，発電機母線，負荷母線に対応するクラスをプログラム**??**からプログラム**??**のように定義しよう。

プログラム 1-7 (bus_slack.m)

```
1  classdef bus_slack
2
3    properties
4      Vabs
5      Vangle
6    end
7
8    methods
9      function obj = bus_slack(Vabs, Vangle)
10       obj.Vabs = Vabs;
11       obj.Vangle = Vangle;
12     end
```

```
13
14      function out = get_constraint(obj, Vr, Vi, P, Q)
15        Vabs = norm([Vr; Vi]);
16        Vangle = atan2(Vi, Vr);
17        out = [Vabs-obj.Vabs; Vangle-obj.Vangle];
18      end
19    end
20  end
```

プログラム 1-8 (bus_genertor.m)

```
1   classdef bus_generator
2
3     properties
4       P
5       Vabs
6     end
7
8     methods
9       function obj = bus_generator(P, Vabs)
10        obj.P = P;
11        obj.Vabs = Vabs;
12      end
13
14      function out = get_constraint(obj, Vr, Vi, P, Q)
15        Vabs = norm([Vr; Vi]);
16        out = [obj.P-P; Vabs-obj.Vabs];
17      end
18    end
19  end
```

プログラム 1-9 (bus_load.m)

```
1   classdef bus_load
2
3     properties
4       P
5       Q
6     end
7
8     methods
9       function obj = bus_load(P, Q)
10        obj.P = P;
```

```
11        obj.Q = Q;
12      end
13
14      function out = get_constraint(obj, Vr, Vi, P, Q)
15        out = [P-obj.P; Q-obj.Q];
16      end
17    end
18  end
```

これらのコードで定義された bus_slack，bus_generator，bus_load はすべて共通の入出力をもつ get_constraint という名前のメソッドを持っている。このとき，これらのクラスを利用するコードは，どの種類の母線であるかを意識せず，get_constraint メソッドを呼べば適切に制約条件を計算することができる。この考え方を用いてプログラム**??**を書き換えると，次のようなプログラムになる。

プログラム 1-10 (func_power_flow.m)

```
1  function [out, V, I, P, Q] = func_power_flow(x, Y, a_bus)
2
3  V = x(1:2:end) + 1j*x(2:2:end);
4  I = Y*V;
5  PQhat = V.*conj(I);
6  P = real(PQhat);
7  Q = imag(PQhat);
8
9  out_cell = cell(numel(a_bus), 1);
10
11 for i = 1:numel(a_bus)
12   bus = a_bus{i};
13   out_cell{i} = bus.get_constraint(...
14     real(V(i)), imag(V(i)), P(i), Q(i));
15 end
16 out = vertcat(out_cell{:});
17
18 end
```

プログラム**??**では，母線の集合であるセル配列が a_bus として入力されることを想定している。この関数の使い方の例示すと，プログラム**??**のようになる。

プログラム 1-11 (main_ex4.m)

```
1  （プログラム 3-6 の 1 行目から 8 行目と同じ）
2
3  a_bus = cell(3, 1);
4  a_bus{1} = bus_slack(2, 0);
5  a_bus{2} = bus_load(-3, 0);
6  a_bus{3} = bus_generator(0.5, 2);
7
8  func_curried = @(x) func_power_flow(x, Y, a_bus);
9
10 x_sol = fsolve(func_curried, x0, options);
11
12 （プログラム 3-4 の 6 行目以下と同じ）
```

プログラム??では，スラック母線，負荷母線，発電機母線をそれぞれ定義し，セル配列 a_bus に代入している。このとき，プログラム??において，a_bus に代入された母線のそれぞれついて get_constraint が実行される。get_constraint は母線の種類ごとに実装されているため，場合分けの記述を行うことなく，それぞれ適切な処理が行われる。このように，同様のコードによる呼び出しに対して，異なる処理が行われることを**多態性**や**ポリモーフィズム**といい，オブジェクト指向プログラミングにおける重要な概念のひとつである。

プログラム??のような多態性を用いた実装では，母線の数や構成が変更された場合への対応が容易であるという利点がある。すなわち，プログラム??の 3 行目から 6 行目における母線の数や定義を変更しさえすれば，プログラム??は変更することなく利用可能である。

さらに，新たな種類の母線の追加も容易である。プログラム??は母線に対して，適切な入出力をもつ get_constraint というメソッドが存在することのみを期待している。言い換えれば get_constraint を持ちさえすれば，それは母線であると定義しているともいえる。これは，「もしそれがアヒルのように歩き，アヒルのように鳴くのなら，それはアヒルである」に由来した**ダックタイピング**という考え方である。このような考え方により，モジュールの設計者はプログラム?? の内部の処理には注意を払うことな

く，新たな母線のクラスを定義し，get_constraint のみを実装すれば良いことがわかる。すなわち，モジュールの設計者が実装を行うべき範囲が明確化されている。

ここまでで，アドミタンス行列 Y が与えられたもとでの潮流計算のプログラムを見通しよく実装することができたといえる。つぎに，アドミタンス行列の計算もオブジェクト指向の考え方に基づきモジュール群に分割して実装してみよう。

例 1.12 (オブジェクト指向によるアドミタンス行列の計算) アドミタンス行列は，複数の送電線によって定義される。そこで，送電線を表すクラスを作成し，そのインスタンスの集合を用いることを考える。送電線は，2 つの母線を結ぶものであるため，それらの母線の番号が特定される必要がある。また，一般に送電線モデルの種類に依らず

$$\begin{bmatrix} \boldsymbol{I}_{\mathrm{from}} \\ \boldsymbol{I}_{\mathrm{to}} \end{bmatrix} = \boldsymbol{Y}_{\mathrm{branch}} \begin{bmatrix} \boldsymbol{V}_{\mathrm{from}} \\ \boldsymbol{V}_{\mathrm{to}} \end{bmatrix} \tag{1.50}$$

を満たすアドミタンス行列 $\boldsymbol{Y}_{\mathrm{branch}}$ が定まる。ただし，$\boldsymbol{V}_{\mathrm{from}}$ と $\boldsymbol{V}_{\mathrm{to}}$ は送電線の両端の母線の電圧であり，$\boldsymbol{I}_{\mathrm{from}}$ と $\boldsymbol{I}_{\mathrm{to}}$ は両端の母線から送電線に流入する電流である。たとえば，例で扱った単純な送電線では

$$\boldsymbol{Y}_{\mathrm{branch}} = \begin{bmatrix} \boldsymbol{y} & -\boldsymbol{y} \\ -\boldsymbol{y} & \boldsymbol{y} \end{bmatrix}$$

となる。

以上のことをまとめると，両端の母線の情報とアドミタンス行列 Y_{branch} を返せばそれを送電線と呼んで良さそうである。この考え方に基づいた送電線の実装例はつぎのようになる。

プログラム 1-12 (branch.m)

```
1  classdef branch
2
```

```
3    properties
4      y
5      from
6      to
7    end
8
9    methods
10     function obj = branch(from, to, y)
11       obj.from = from;
12       obj.to = to;
13       obj.y = y;
14     end
15
16     function Y = get_admittance_matrix(obj)
17       y = obj.y;
18       Y = [y, -y;
19         -y,  y];
20     end
21   end
22 end
```

この送電線クラスを用いてアドミタンス行列を計算する関数を実装すると，つぎのようになる。

プログラム 1-13 (get_admittance_matrix.m)

```
1  function Y = get_admittance_matrix(n_bus, a_branch)
2
3  Y = zeros(n_bus, n_bus);
4
5  for i = 1:numel(a_branch)
6    br = a_branch{i};
7    Y_branch = br.get_admittance_matrix();
8    Y([br.from, br.to], [br.from, br.to]) =...
9      Y([br.from, br.to], [br.from, br.to]) + Y_branch;
10 end
11
12 end
```

このプログラムの入力引数 n_bus は母線の数を表し，a_branch は送電線の入ったセル配列である。このプログラムはたとえばプログラム**??**のように使用することができる。

プログラム 1-14 (main_admittance_matrix.m)

```
1  a_branch = cell(2, 1);
2  a_branch{1} = branch(1, 2, 1.3652-11.6040j);
3  a_branch{2} = branch(2, 3, -10.5107j);
4
5  Y = get_admittance_matrix(3, a_branch)
```

プログラム**??**では，送電線が from と to というデータを持つことと，get_admittance_matrix というメソッドを持つことのみを期待している。したがって，π 型モデルなどの他の送電線モデルを考える場合にも，to，from という変数と，get_admittance_matrix というメソッドをもつクラスを実装するだけで良く，プログラム**??**は変更することなく利用することができる。

最後に，以上のプログラムをまとめて，潮流計算を行うプログラムとする。

例 1.13 (潮流計算の実装結果) ここまでの例で実装したプログラムをまとめて，潮流計算を行う関数として整理すると，プログラム**??**となる。

プログラム 1-15 (calculate_power_flow.m)

```
1  function [V, I, P, Q] = calculate_power_flow(a_bus, a_branch)
2
3  n_bus = numel(a_bus);
4  Y = get_admittance_matrix(n_bus, a_branch);
5
6  func_curried = @(x) func_power_flow(x, Y, a_bus);
7
8  x0 = kron(ones(n_bus, 1), [1; 0]);
9  options = optimoptions('fsolve', 'Display', 'iter');
10 x_sol = fsolve(func_curried, x0, options);
11
12 [~, V, I, P, Q] = func_curried(x_sol);
13
14 end
```

このプログラムは母線の集合と送電線の集合を受け取って潮流計算を行うものであり，プログラム**??**のように利用することができる。

プログラム 1-16 (main_power_flow.m)

```
1  a_bus = cell(3, 1);
2  a_bus{1} = bus_slack(2, 0);
3  a_bus{2} = bus_load(-3, 0);
4  a_bus{3} = bus_generator(0.5, 2);
5
6  a_branch = cell(2, 1);
7  a_branch{1} = branch(1, 2, 1.3652-11.6040j);
8  a_branch{2} = branch(2, 3, -10.5107j);
9
10 [V, I, P, Q] = calculate_power_flow(a_bus, a_branch);
11
12 display('Vabs:'), display(abs(V)')
13 display('Vangle:'), display(angle(V)')
14 display('P:'), display(P')
15 display('Q:'), display(Q')
```

異なるネットワーク構造に対して潮流計算を行う必要がある場合も，プログラム**??**の 1 行目から 8 行目において母線や送電線の定義のみ変えれば，他のプログラムは全く変更することなく利用することができる。また，新たな母線や送電線の種類が必要になった場合でも，それぞれ決められた名前のデータやメソッドを実装した新たなクラスを実装すればよい。これらのことにより，例**??**のようなシンプルな実装と比較して，可読性や拡張性が高いプログラムとなっている。

1.7 電力系統の時間応答シミュレーションの実装法

本節では，MATLAB を用いて電力システムの時間応答をシミュレーションするプログラムを実装する方法について説明する。電力系統の時間応答のシミュレーションは，機器の動特性を表す微分方程式と，電力潮流の代数方程式を連立した微分代数方程式を解くことによって行われる。そこで，まずは簡単な微分代数方程式を例に，その解き方を見てみよう。

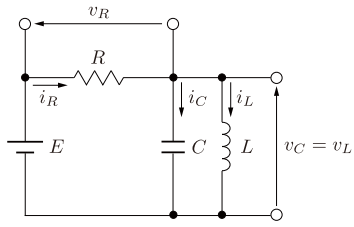


図 1.12 微分代数方程式の例題：LC 並列回路

例 1.14 (簡単な例題) **??**の電気回路において，$R = L = C = E = 1$ のときのシミュレーションを行うことを考える。この回路の動的な要素はコイル L とコンデンサ C であり，その微分方程式は，

$$L\dot{i}_L = v_L \tag{1.51a}$$

$$C\dot{v}_C = i_C \tag{1.51b}$$

である。ただし，初期値は $i_L(0) = 0$, $v_C(0) = 0$ とする。また，オームの法則とキルヒホッフの法則から，

$$v_R = Ri_R \tag{1.52a}$$

$$i_R = i_L + i_C \tag{1.52b}$$

$$v_L = v_C \tag{1.52c}$$

$$E = v_C + v_R \tag{1.52d}$$

という代数方程式が成り立つ。

このシステムは微分方程式と代数方程式が連立された微分代数方程式である。式**??**の代数方程式を用いて文字を消去すると，等価な常微分方程式を得ることができる。これはクロン縮約に対応し，得られる常微分方程式は，

$$\dot{i}_L = \frac{1}{L} v_C \tag{1.53a}$$

$$\dot{v}_C = \frac{1}{RC}(E - v_C) - \frac{1}{C} i_L \tag{1.53b}$$

となる。まずは，この常微分方程式を解くプログラムを書いてみよう。常備分方程式に対する MATLAB のソルバとしては ode45 を用いるのが一般的である。ode45 では，常微分方程式 $\dot{x} = f(t, x)$ の関数 $f(t, x)$ を実装することにより常微分方程式を解くことができる。式 (**??**) の右辺を実装すると，プログラム**??**のようになる。

プログラム 1-17 (func_RLC_ode.m)

```
1  function dx = func_RLC_ode(x, R, C, L, E)
2
3  iL = x(1);
4  vC = x(2);
5
6  diL = vC/L;
7  dvC = (E-vC)/R/C - iL/C;
8
9  dx = [diL; dvC];
10
11 end
```

また，これを用いて ode45 を実行し，常微分方程式を解くプログラムはプログラム**??**となる。

プログラム 1-18 (main_RLC_ode.m)

```
1  R = 1;
2  L = 1;
3  C = 1;
4  E = 1;
5
6  func = @(t, x) func_RLC_ode(x, R, C, L, E);
7  x0 = [0; 0];
8  tspan = [0 30];
9
10 [t, x] = ode45(func, tspan, x0);
11
12 plot(t, x)
```

このプログラムにおいて，出力された変数 x は，i_L と v_C の時系列を並べたベクトルである。したがって，i_C や i_R などの他の変数については式(**??**) の代数方程式を用いて改めて計算する必要があることに注意しよう。

つぎに，クロン縮約を行わず，微分代数方程式を直接解くことを考える。これには，物理的な代数方程式をそのまま用いることとができ，複雑なシステムになっても記述が簡単である利点がある。MATLAB で微分代数方程式を解くことができるコマンドのひとつに ode15s がある。ode15s は

$$M\dot{x} = f(t, x)$$

という形で記述された微分代数方程式を対象とする。式**??**，**??**をこの形にあてはめると，

$$\begin{bmatrix} 0 & L & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & C \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{i}_R \\ \dot{i}_L \\ \dot{i}_C \\ \dot{v}_R \\ \dot{v}_L \\ \dot{v}_C \end{bmatrix} = \begin{bmatrix} v_L \\ i_C \\ v_R - Ri_R \\ i_R - i_L - i_C \\ v_L - v_C \\ E - v_C - v_R \end{bmatrix}$$

となる。ここで，右辺の 3 番目以降の要素の順番は任意であることに注意しよう。この右辺を実装するとプログラム**??**となる。

プログラム 1-19 (func_RLC_dae.m)

```
1  function dx = func_RLC_dae(x, R, C, L, E)
2
3  iR = x(1);
4  iL = x(2);
5  iC = x(3);
6  vR = x(4);
7  vL = x(5);
8  vC = x(6);
9
10 diL = vL;
11 dvC = iC;
12
```

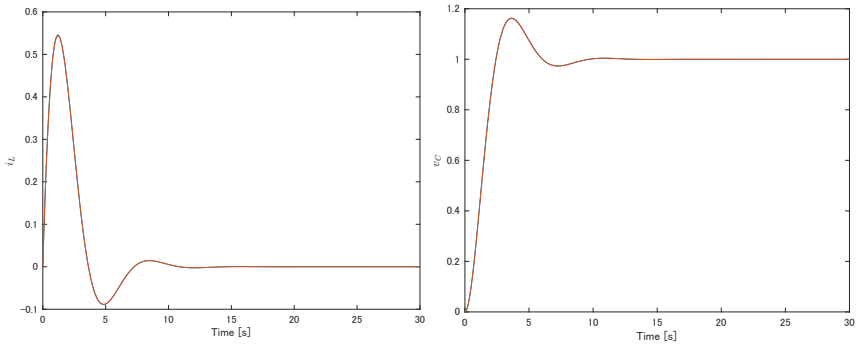
```
13  con1 = vC-vL;
14  con2 = E-vC-vR;
15  con3 = iR-(iC+iL);
16  con4 = vR-iR*R;
17
18  dx = [diL; dvC; con1; con2; con3; con4];
19  end
```

この関数を用いると，プログラム**??**のように微分代数方程式を解くことができる。

プログラム 1-20 (main_RLC_dae.m)

```
1  R = 1;
2  C = 1;
3  L = 1;
4  E = 1;
5
6  M = zeros(6, 6);
7  M(1, 2) = L;
8  M(2, 6) = C;
9
10 x0 = zeros(6, 1);
11 tspan = [0 30];
12
13 options = odeset('Mass', M);
14 [t, x] = ode15s(@(t, x) func_RLC_dae(x, R, C, L, E),...
15   tspan, x0, options);
16
17 plot(t, x(:, [2, 6]))
```

このプログラムにおいて，13 行目のオプションで左辺の M を設定している。また，10 行目で初期状態を設定しているが，意味を持つのは動的な要素の値のみ，すなわち，2 番目と 6 番目の要素のみである。他の値については代数方程式を満たす値が `ode15s` によって探索されるため，代数方程式を満たす値を計算して設定する必要はない。このプログラムでは x の初期値をすべての要素を 0 としたベクトルとしており，実際，この場合は代数方程式は満たされないが，問題なく解が求まる。ここで，15 行目において得られる解 `x` は，$v_R, v_L, v_C, i_R, i_L, i_C$ の時系列を並べた行列である。



(a) i_L の時間応答　　(b) v_C の時間応答

図 1.13　LC 並列回路の時間応答

常微分方程式に変換した場合と異なり，すべての物理量を含んでいる。

常微分方程式として解いた場合と微分代数方程式として解いた場合の i_L と v_C を?? に示す。それぞれの図には 2 本の線が表示してあるが，当然ながらこれらは完全に重なっている。

1.7.1　電力系統のシミュレーションのシンプルな実装

本項では, 電力系統のシミュレーションをシンプルに実装する方法を説明する。

例 1.15　(電力系統のシミュレーションのシンプルな実装)　例??で扱った電力系統モデルの時間応答をシミュレーションするプログラムを実装しよう。このシステムを記述するためには, 発電機が接続された母線 $i \in \{1, 3\}$ について，式??の微分方程式が必要であり，また，式??の代数方程式が成り立っている。母線 2 には定インピーダンス負荷が接続されており，式?? の代数方程式が満たされている。さらにネットワーク全体でキルヒホッフの法則である式??の代数方程式が成り立つ。これらの式で用いられる変数をまとめて,

$$x = \left[\delta_1, \Delta\omega_1, E_1, \delta_3, \Delta\omega_3, E_3, \boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3, \boldsymbol{I}_1, \boldsymbol{I}_2, \boldsymbol{I}_3\right]^{\mathsf{T}}$$

とおき，$M\dot{x} = f(t,x)$ の形で記述することを考えると，右辺 $f(t,x)$ の実装例はプログラム?? となる．

プログラム 1-21 (func_simulation_3bus.m)

```
function dx = func_simulation_3bus(x, Y, parameter)

delta1 = x(1);
omega1 = x(2);
E1 = x(3);
delta3 = x(4);
omega3 = x(5);
E3 = x(6);
V1 = x(7)  + 1j*x(8);
V2 = x(9)  + 1j*x(10);
V3 = x(11) + 1j*x(12);
I1 = x(13) + 1j*x(14);
I2 = x(15) + 1j*x(16);
I3 = x(17) + 1j*x(18);

omega0 = parameter.omega0;

X1 = parameter.X1;
X1_prime = parameter.X1_prime;
M1 = parameter.M1;
D1 = parameter.D1;
tau1 = parameter.tau1;
Pmech1 = parameter.Pmech1;
Vfield1 = parameter.Vfield1;

z2 = parameter.z2;

X3 = parameter.X3;
X3_prime = parameter.X3_prime;
M3 = parameter.M3;
D3 = parameter.D3;
tau3 = parameter.tau3;
Pmech3 = parameter.Pmech3;
Vfield3 = parameter.Vfield3;

P1 = real(V1*conj(I1));
P3 = real(V3*conj(I3));

```

```
39  dx1 = [omega0 * omega1;
40      (-D1*omega1-P1+Pmech1)/M1;
41      (-X1/X1_prime*E1+...
42      (X1/X1_prime-1)*abs(V1)*cos(delta1-angle(V1))+Vfield1)/tau1];
43
44  dx3 = [omega0 * omega3;
45      (-D3*omega3-P3+Pmech3)/M3;
46      (-X3/X3_prime*E3+...
47      (X3/X3_prime-1)*abs(V3)*cos(delta3-angle(V3))+Vfield3)/tau3];
48
49  con1 = I1-(E1*exp(1j*delta1)-V1)/(1j*X1_prime);
50  con2 = V2+z2*I2;
51  con3 = I3-(E3*exp(1j*delta3)-V3)/(1j*X3_prime);
52
53  con_network = [I1; I2; I3] - Y*[V1; V2; V3];
54
55  dx = [dx1; dx3;real(con1); imag(con1); real(con2);
56      imag(con2); real(con3); imag(con3);
57      real(con_network); imag(con_network)];
58
59  end
```

このプログラムにおいて，x に含まれる電圧や電流は実部と虚部を並べたものとして表現している。また，変数をまとめて指定するために構造体 parameter を利用している。このプログラムを用いて微分代数方程式を解くプログラムはプログラム**??**となる。

プログラム 1-22 (main_simulation_simple.m)

```
1   a_branch = cell(2, 1);
2   a_branch{1} = branch(1, 2, 1.3652-11.6040j);
3   a_branch{2} = branch(2, 3, -10.5107j);
4   Y = get_admittance_matrix(3, a_branch);
5
6   parameter = struct();
7
8   parameter.M1 = 100;
9   parameter.D1 = 10;
10  parameter.tau1 = 5.14;
11  parameter.X1 = 1.569;
12  parameter.X1_prime = 0.936;
13  parameter.Pmech1 = 2.5158;
14  parameter.Vfield1 = 2.7038;
15
```

```
16  parameter.z2 = 1.3224;
17
18  parameter.M3 = 12;
19  parameter.D3 = 10;
20  parameter.tau3 = 8.97;
21  parameter.X3 = 1.220;
22  parameter.X3_prime = 0.667;
23  parameter.Pmech3 = 0.5000;
24  parameter.Vfield3 = 2.1250;
25
26  parameter.omega0 = 60*2*pi;
27
28  x0 = [0.5357 + pi/6; 0; 2.3069 + 0.1;...
29      0.0390; 0; 2.0654; zeros(12, 1)];
30  M = blkdiag(eye(6), zeros(12, 12));
31
32  tspan = [0 50];
33
34  options = odeset('Mass', M);
35  func = @(t, x) func_simulation_3bus(x, Y, parameter);
36  [t, x] = ode15s(func, tspan, x0, options);
37
38  plot(t, x(:, [2, 5]))
```

プログラム**??**を実行すると，電力系統の初期値応答を計算することができ，発電機の周波数偏差がプロットされる。

1.7.2 分割したモジュール群を用いた時間応答シミュレーションの実装法

本項では，前項で述べたプログラムを機能ごとに分離し，拡張性の高いプログラムに変更する方法について説明する。

例 1.16 (発電機と負荷のモジュール化) プログラム**??**は，入力された x をそれぞれの機器の状態変数，電圧，電流に分割する処理とそれらを用いて微分方程式や代数方程式の計算を行う処理という 2 つの要素から成り立っている。これらの 2 つの要素を行うことができるプログラムを見通しよく記述する方法を考えよう。

変数 x を適切に分割するためには，それぞれの機器の状態の数を把握する必要がある。また，すべての機器について，微分方程式のための状態の時間微分 $\mathrm{d}x/\mathrm{d}t$ と，代数方程式 $f(x) = 0$ における $f(x)$ の計算が行われている。ダックタイピングの考え方を用いれば，状態数を返す機能と時間微分，代数方程式の計算を行う機能を有していればそれを機器と呼んでよさそうである。これらの機能を有する機器として発電機と負荷を実装すると，それぞれプログラム**??**と**??**のようになる。

プログラム 1-23 (generator.m)

```
1  classdef generator < handle
2
3  properties
4    omega0
5    X
6    X_prime
7    M
8    D
9    tau
10   Pmech
11   Vfield
12 end
13
14 methods
15   function obj = generator(omega0, M, D, tau,...
16       X, X_prime, Pmech, Vfield)
17
18     obj.omega0 = omega0;
19     obj.X = X;
20     obj.X_prime = X_prime;
21     obj.M = M;
22     obj.D = D;
23     obj.tau = tau;
24     obj.Pmech = Pmech;
25     obj.Vfield = Vfield;
26   end
27
28   function nx = get_nx(obj)
29     nx = 3;
30   end
31
32   function [dx, con] = get_dx_constraint(obj, x, V, I)
33     delta = x(1);
```

```
34      omega = x(2);
35      E = x(3);
36      P = real(V*conj(I));
37
38      Pmech = obj.Pmech;
39      Vfield = obj.Vfield;
40
41      X = obj.X;
42      X_prime = obj.X_prime;
43      D = obj.D;
44      M = obj.M;
45      tau = obj.tau;
46
47      omega0 = obj.omega0;
48
49      dE = (-X/X_prime*E+...
50        (X/X_prime-1)*abs(V)*cos(delta-angle(V))...
51        +Vfield)/tau;
52      dx = [omega0 * omega;
53        (-D*omega-P+Pmech)/M;
54        dE];
55      con = I-(E*exp(1j*delta)-V)/(1j*X_prime);
56      con = [real(con); imag(con)];
57    end
58  end
59
60  end
```

プログラム 1-24 (load_impedance.m)

```
1   classdef load_impedance < handle
2
3   properties
4     z
5   end
6
7   methods
8     function obj = load_impedance(z)
9       obj.z = z;
10    end
11
12    function nx = get_nx(obj)
13      nx = 0;
14    end
15
```

```
16    function [dx, con] = get_dx_constraint(obj, x, V, I)
17      dx = [];
18      z = obj.z;
19      con = V+z*I;
20      con = [real(con); imag(con)];
21    end
22  end
23
24  end
```

これらのプログラムでは，メソッド get_nx が状態変数の数を返し，get_dx_constraint が状態変数の時間微分と制約条件を返す。このように定義された機器を用いて，プログラム?? はプログラム??のように書き換えることができる。

プログラム 1-25 (func_simulation.m)

```
1   function out = func_simulation(t, x, Y, a_component)
2
3   n_component = numel(a_component);
4   x_split = cell(n_component, 1);
5   V = zeros(n_component, 1);
6   I = zeros(n_component, 1);
7
8   idx = 0;
9   for k = 1:n_component
10    nx = a_component{k}.get_nx();
11    x_split{k} = x(idx+(1:nx));
12    idx = idx + nx;
13  end
14
15  for k = 1:n_component
16    V(k) = x(idx+1) + x(idx+2)*1j;
17    idx = idx + 2;
18  end
19
20  for k = 1:n_component
21    I(k) = x(idx+1) + x(idx+2)*1j;
22    idx = idx + 2;
23  end
24
25  dx = cell(n_component, 1);
26  con = cell(n_component, 1);
27
```

```
28  for k = 1:n_component
29    component = a_component{k};
30    xk = x_split{k};
31    Vk = V(k);
32    Ik = I(k);
33    [dx{k}, con{k}] = component.get_dx_constraint(xk, Vk, Ik);
34  end
35
36  con_network = I - Y*V;
37
38  out = vertcat(dx{:});
39  out = [out; vertcat(con{:})];
40  out = [out; real(con_network); imag(con_network)];
41
42  end
```

このプログラムでは，a_component に機器のセル配列が入力されることを期待している。そして，9 行目から 23 行目で変数 x の分割を行っている。このとき，10 行目で機器の状態の数を取得することにより，適切に分割を行うことができる。また，28 行目から 34 行目において，状態の時間微分と制約条件を計算する。プログラムの 33 行目では，get_dx_constraint を呼び出しており，これは多態性を用いた実装となっている。プログラムの 38 行目の vertcat(dx{:}) は，セル配列 dx の要素をすべて垂直方向に結合している。すなわち，[dx{1}; dx{2}; ...; dx{end}] と等価である。

プログラム**??**を用いてシミュレーションを実行することを考え，プログラム**??**のシミュレーションに関する部分をまとめると，プログラム**??** のようになる。

プログラム 1-26 (simulate_power_system.m)

```
1  function [t, x, V, I] = ...
2    simulate_power_system(a_component, Y, x0, tspan)
3
4  n_component = numel(a_component);
5  a_nx = zeros(n_component, 1);
6  for k = 1:n_component
7    component = a_component{k};
```

```
 8    a_nx(k) = component.get_nx();
 9  end
10  nx = sum(a_nx);
11
12  M = blkdiag(eye(nx), zeros(n_component*4));
13  options = odeset('Mass', M, 'RelTol', 1e-6);
14
15  y0 = [x0(:); zeros(4*n_component, 1)];
16
17  [t, y] = ode15s(@(t, x) func_simulation(t, x, Y, a_component),...
18    tspan, y0, options);
19
20
21  x = cell(n_component, 1);
22  V = zeros(numel(t), n_component);
23  I = zeros(numel(t), n_component);
24
25  idx = 0;
26
27  for k = 1:n_component
28    x{k} = y(:, idx+(1:a_nx(k)));
29    Vk = y(:, nx+2*(k-1)+(1:2));
30    V(:, k) = Vk(:, 1) + 1j*Vk(:, 2);
31    Ik = y(:, nx+2*n_component+2*(k-1)+(1:2));
32    I(:, k) = Ik(:, 1) + 1j*Ik(:, 2);
33    idx = idx + a_nx(k);
34  end
35
36  end
```

プログラム**??**では，5 行目から 9 行目でそれぞれの機器から得た状態の数を用いて，行列 M を定義している。そして，プログラム**??** を用いて微分代数方程式を解き，27 行目から 34 行目において，結果をそれぞれの機器の状態変数，それぞれの母線の電圧と電流に分割して返している。

さらに，プログラム**??**を書き換えると，プログラム**??** のようになる。

プログラム 1-27 (main_simulation_3bus.m)

```
1  a_branch = cell(2, 1);
2  a_branch{1} = branch(1, 2, 1.3652-11.6040j);
3  a_branch{2} = branch(2, 3, -10.5107j);
4
5  Y = get_admittance_matrix(3, a_branch);
```

```
6
7  gen1 = generator(60*2*pi, 100, 10, 5.14,...
8    1.569, 0.936, 2.5158, 2.7038);
9
10 load2 = load_impedance(1.3224);
11
12 gen3 = generator(60*2*pi, 12, 10, 8.97,...
13   1.220, 0.667, 0.5000, 2.1250);
14
15 a_component = {gen1; load2; gen3};
16
17 x0 = [0.5357 + pi/6; 0; 2.3069 + 0.1; 0.0390; 0; 2.0654];
18 tspan = [0 50];
19
20 [t, x, V, I] = simulate_power_system(a_component, Y, x0, tspan);
21
22 plot(t, [x{1}(:, 2), x{3}(:, 2)])
```

プログラム??では，発電機や負荷を表す機器の配列 a_component を作成し，プログラム??で定義した関数 simulate_power_flow を適用している。時間応答のシミュレーションにおいて，ネットワークの構造や利用する機器を変更する場合にはプログラム??の a_branch と a_component を変更するのみで良く，プログラム??から??を変更する必要はない。また，プログラム??や??の発電機や負荷とは別の機器を用いる場合には get_nx と get_dx_constraint をもつクラスを実装して用いるだけで良い。これらのことにより，例??の実装と比較して，可読性や拡張性が高いプログラムとなっている。

例??では，発電機や負荷のモジュール化によって電力システムの時間応答を見通し良く実装することができた。しかし，プログラム??では，発電機の V_{field} や P_{mech}，負荷の z はシミュレーションにおける所望の平衡点に整合する値をあらかじめ計算しておく必要がある。これらの値はそれぞれの機器の動特性に異存するため，機能の分離の観点からは不十分である。そこで，例??で述べたプログラムを拡張し，機能的により分離したプログラムとすることを考えよう。

例 1.17 電力系統の時間応答のシミュレーションにおいて，所望の電力供給を実現する定常状態を達成することを考える。このとき，発電機と負荷のそれぞれにおいて，式??と式??が成り立っている必要がある。これらの関係式はそれぞれの機器の状態方程式と関連づいているため，generator や load_impedance クラスに実装するのが適切である。この観点から，プログラム??と??に所望の電力消費を達成する定常状態を計算するメソッドを追加するとプログラム??と??のようになる。

プログラム 1-28 (generator.m)

```
1  classdef generator < handle
2
3  （プログラム 3-23 の 3 行目から 12 行目と同じ）
4
5  methods
6
7  （プログラム 3-23 の 15 行目から 57 行目と同じ）
8
9    function x_equilibrium = set_equilibrium(obj, V, I, P, Q)
10     Vabs = abs(V);
11     Vangle = angle(V);
12
13     X = obj.X;
14     X_prime = obj.X_prime;
15
16     delta = Vangle + atan(P/(Q+Vabs^2/X_prime));
17     E = X_prime/Vabs*sqrt((Q+Vabs^2/X_prime)^2+P^2);
18
19     x_equilibrium = [delta; 0; E];
20
21     obj.Pmech = P;
22     obj.Vfield = X*E/X_prime ...
23       - (X/X_prime-1)*Vabs*cos(delta-Vangle);
24   end
25
26 end
27
28 end
```

プログラム 1-29 (load_impedance.m)

```
1  classdef load_impedance < handle
2
3  （プログラム 3-24 の 3 行目から 5 行目と同じ）
4
5  methods
6
7  （プログラム 3-24 の 8 行目から 21 行目と同じ）
8
9    function x_equilibrium = set_equilibrium(obj, V, I, P, Q)
10     x_equilibrium = [];
11     obj.z = -V/I;
12   end
13
14 end
15
16 end
```

これらのプログラムにおいて，set_equilibrium は所望の定常状態における電圧 V，電流 I，電力 P を受け取り，定常状態 x_equilibrium を返すメソッドである。ここで，理論的には電流 I を与える必要はなく，V と P のみで十分であるが，利便性のために I も与えている。潮流計算で得られたパラメータを用いて例??と同様のシミュレーションを行うプログラムは??のように記述することができる。

プログラム 1-30 (main_simulation_3bus_equilibrium.m)

```
1  a_bus = cell(3, 1);
2  a_bus{1} = bus_slack(2, 0);
3  a_bus{2} = bus_load(-3, 0);
4  a_bus{3} = bus_generator(0.5, 2);
5
6  a_branch = cell(2, 1);
7  a_branch{1} = branch(1, 2, 1.3652-11.6040j);
8  a_branch{2} = branch(2, 3, -10.5107j);
9
10 gen1 = generator(60*2*pi, 100, 10, 5.14, 1.569, 0.936, [], []);
11 load2 = load_impedance([]);
12 gen3 = generator(60*2*pi, 12, 10, 8.97, 1.220, 0.667, [], []);
13
14 a_component = {gen1; load2; gen3};
15
```

```
16  [V, I, P, Q] = calculate_power_flow(a_bus, a_branch);
17  Y = get_admittance_matrix(3, a_branch);
18
19  x_equilibrium = cell(numel(a_component), 1);
20  for k=1:numel(a_component)
21     x_equilibrium{k} =...
22       a_component{k}.set_equilibrium(V(k), I(k), P(k), Q(k));
23  end
24
25  x0 = vertcat(x_equilibrium{:});
26  x0(1) = x0(1) + pi/6;
27  x0(3) = x0(3) + 0.1;
28  tspan = [0 50];
29
30  [t, x, V, I] = simulate_power_system(a_component, Y, x0, tspan);
31
32  plot(t, [x{1}(:, 2), x{3}(:, 2)])
```

このプログラムでは, 母線, 送電線, 発電機, 負荷のクラスを利用して電力系統を定義し, 潮流計算を行い, 時間応答を計算するという一連の流れが一目瞭然である。ユーザはそれぞれの要素の物理パラメータを指定さえすれば, 内部の動特性などについては注意を払う必要なくシミュレーションを実行することができる。この例においては機器のモデルが `set_equilibrium` というメソッドをもつことを期待している。これは, ダックタイピングにおける機器の定義を変更していることになる。

ここまでの例では電力系統の初期状態応答のシミュレーションを扱ってきた。つぎに, 地絡に対する応答の計算の実装法について述べる。

例 1.18 (地絡に対する応答のシミュレーション) 地絡に対する応答を計算するためには, **??**節で述べたように, 地絡の生じた母線の電圧を一定時間 0 に固定すれば良い。これは, プログラム**??**の 36 行目の制約条件を一部変更することによってつぎのように達成される。

プログラム 1-31 (func_simulation.m)

```
1  function out = func_simulation(x, Y, a_component, bus_fault)
```

```
2
3  （プログラム 3-25 の 3 行目から 34 行目と同じ）
4
5  con_network = I - Y*V;
6  con_network(bus_fault) = V(bus_fault);
7
8  （プログラム 3-25 の 38 行目から 40 行目と同じ）
9
10 end
```

このプログラムでは，入力引数が追加されており，bus_fault に地絡が生じる母線の番号が代入されることが想定されている。このプログラムを用いるためにプログラム??を変更すると，??のようになる。

プログラム 1-32 (simulate_power_system.m)

```
1  function [t, x, V, I] = simulate_power_system(a_component, ...
2    Y, x0, tspan,bus_fault, tspan_fault)
3
4  if nargin < 5
5    bus_fault = [];
6  end
7
8  if nargin < 6
9    tspan_fault = [0, 0];
10 end
11
12 （プログラム 3-26 の 4 行目から 15 行目と同じ）
13
14 if isempty(bus_fault)
15   [t, y] = ode15s(...
16     @(t, x) func_simulation(t, x, Y, a_component, []),...
17     tspan, y0, options);
18 else
19   [t1, y1] = ode15s(...
20     @(t, x) func_simulation(t, x, Y, a_component, bus_fault),...
21   tspan_fault, y0, options);
22
23   [t2, y2] = ode15s(...
24     @(t, x) func_simulation(t, x, Y, a_component, []),...
25   [tspan_fault(2), tspan(2)], y1(end, :), options);
26
27   t = [t1; t2];
28   y = [y1; y2];
```

```
29  end
30
31  （プログラム 3-26 の 21 行目から 34 行目と同じ）
32
33  end
```

このプログラムでは入力引数に地絡が生じる母線とその期間が追加されている。ただし 4 行目から 10 行目においてこれらの値が入力されなかった場合のデフォルト値を設定しているため，プログラム**??**と同様に地絡を指定せずに利用することもできる。このように，過去のシステム向けのコードが新しいシステムでも利用できることを**後方互換性**という。

プログラム**??**では，19 行目から 21 行目において地絡が生じているとき，23 行目から 25 行目において地絡が解消された後のシミュレーションを実行している。このとき，地絡が解消された後のシミュレーションにおける初期状態は，地絡が生じた場合のシミュレーションにおける最終値としていることに注意が必要である。このプログラムを用いて地絡に対する応答をシミュレーションするプログラムを記述すると，プログラム**??**のようになる。

プログラム 1-33 (main_simulation_3bus_fault.m)

```
1
2  （プログラム 3-30 の 1 行目から 23 行目と同じ）
3
4  x0 = vertcat(x_equilibrium{:});
5  tspan = [0 50];
6
7  fault_bus = 1;
8  fault_tspan = [0, 50e-3];
9
10 [t, x, V, I] = simulate_power_system(a_component, Y, x0, tspan,...
11   fault_bus, fault_tspan);
12
13 plot(t, [x{1}(:, 2), x{3}(:, 2)])
```

このプログラムを実行すると，**??**に相当するプロットを得ることがで

きる。

時間応答のシミュレーションの実装法の最後に，入力応答の計算法を述べる。

例 1.19 これまで扱ってきた例において，発電機のパラメータ P_{mech} を変動させるような入力信号を印加することや，例**??**のように負荷の大きさを変化させたシミュレーションを行うことを考える。このために，外部入力を受け入れることができるようにプログラムを変更する。

外部入力を考慮してシミュレーションを行うためには，各機器が受け取ることができる入力の数が明示されていることが必要である。この観点から，機器の定義に「入力の数を返すメソッドがあること」を追加する。さらに，状態変数の時間微分や制約条件の計算に外部入力を反映させることを考えて**??**と**??**を修正すると，**??**と**??**のようになる。

プログラム 1-34 (generator.m)

```
1  classdef generator < handle
2
3   （プログラム 3-23 の 3 行目から 12 行目と同じ）
4
5  methods
6
7   （プログラム 3-23 の 15 行目から 57 行目と同じ）
8
9    function nu = get_nu(obj)
10     nx = 1;
11   end
12
13   function [dx, con] = get_dx_constraint(obj, x, V, I, u)
14
15  （プログラム 3-23 の 33 行目から 36 行目と同じ）
16
17     Pmech = obj.Pmech + u;
18
19  （プログラム 3-23 の 39 行目から 56 行目と同じ）
20
21   end
22
23 （プログラム 3-28 の 9 行目から 24 行目と同じ）
```

```
24
25  end
26  end
```

プログラム 1-35 (load_impedance.m)

```
1   classdef load_impedance < handle
2
3    （プログラム 3-24 の 3 行目から 5 行目と同じ）
4
5   methods
6
7    （プログラム 3-24 の 8 行目から 21 行目と同じ）
8
9     function nu = get_nu(obj)
10      nu = 2;
11    end
12
13    function [dx, con] = get_dx_constraint(obj, x, V, I, u)
14      dx = [];
15      z = real(obj.z)*(1+u(1)) + 1j*imag(obj.z)*(1+u(2));
16      con = V+z*I;
17      con = [real(con); imag(con)];
18    end
19
20   （プログラム 3-29 の 9 行目から 12 行目と同じ）
21
22  end
23
24  end
```

これらプログラムでは，get_nu が受け取ることができる入力の数を返している。また，get_dx_constraint が入力 u を受け取って処理するように変更されている。このとき，発電機に対する入力は P_{mech} の増分を表し，負荷に対する入力はインピーダンスの実部，虚部の変化の割合を表わしている。これらの機器を使って入力応答が計算できるように**??**と**??** を修正すると**??**と**??**となる。

プログラム 1-36 (func_simulation.m)

```
1   function out = func_simulation(t, x, Y, a_component,...
```

```
 2    bus_fault, U, bus_U)
 3
 4  （プログラム 3-25 の 3 行目から 26 行目と同じ）
 5
 6  for k = 1:n_component
 7
 8  （プログラム 3-25 の 29 行目から 32 行目と同じ）
 9
10    if ismember(k, bus_U)
11      uk = U{bus_U==k}(t);
12    else
13      uk = zeros(component.get_nu(), 1);
14    end
15
16    [dx{k}, con{k}] = component.get_dx_constraint(xk, Vk, Ik, uk);
17
18  end
19
20  （プログラム 3-31 の 5 行目から 6 行目と同じ）
21
22  （プログラム 3-25 の 38 行目から 40 行目と同じ）
23
24  end
```

プログラム 1-37 (simulate_power_system.m)

```
 1  function [t, x, V, I] = simulate_power_system(a_component, ...
 2    Y, x0, tspan,bus_fault, tspan_fault, U, bus_U)
 3
 4  （プログラム 3-32 の 4 行目から 10 行目と同じ）
 5
 6  if nargin < 7
 7      U = {};
 8      bus_U = [];
 9  end
10
11  （プログラム 3-26 の 4 行目から 15 行目と同じ）
12
13  if isempty(bus_fault)
14    [t, y] = ode15s(...
15      @(t, x) func_simulation(t, x, Y, a_component, [], U, bus_U),...
16      tspan, y0, options);
17  else
18    [t1, y1] = ode15s(...
```

```
19      @(t, x) func_simulation(t, x, Y, a_component, bus_fault, U, bus_U),...
20    tspan_fault, y0, options);
21
22    [t2, y2] = ode15s(...
23      @(t, x) func_simulation(t, x, Y, a_component, [], U, bus_U),...
24    [tspan_fault(2), tspan(2)], y1(end, :), options);
25
26    t = [t1; t2];
27    y = [y1; y2];
28  end
29
30   （プログラム 3-26 の 21 行目から 34 行目と同じ）
31
32  end
```

これらのプログラムでは入力引数に U と bus_U が追加されている。ここで，bus_U には入力信号を指定する母線の番号が代入されていることを想定している。また，U は指定された母線の数の要素をもつセル配列であり，その要素は時刻 t を受け取り，入力信号を返す関数である。

変更された関数を用いてシミュレーションを行うプログラムを例示すると，プログラム**??**

プログラム 1-38 (main_simulation_3bus_input.m)

```
1
2     （プログラム 3-30 の 1 行目から 25 行目と同じ）
3
4   tspan = [0 50];
5
6   bus_U = [1; 2];
7   U = {@(t) 0; @(t) [0.05*t/50; 0.05*t/50]};
8
9   [t, x, V, I] = simulate_power_system(a_component, Y, x0, tspan, [], [], U, bus_U);
10
11  plot(t, [x{1}(:, 2), x{3}(:, 2)])
12
```

このプログラムでは TODO 行目から TOOD 行目において入力信号を定義している。このプログラムでは，負荷のインピーダンスを 50 秒間かけて 5%増加させている。母線 1 の入力信号は常に 0 を返すものであり意

味を持たないが，指定の方法を例示するために追加した。また，母線 3 については入力を指定していないが，このような場合には自動的に 0 が入力される。以上のようなプログラムの変更により，外部入力に対するシミュレーションを行うことができる。この例で行った変更は後方互換性を保つように留意されているため，プログラム**??**を適切に書き換えることで，さまざまな電力系統に対する初期値応答，地絡に対する応答，入力応答を計算することができる。このとき，プログラム**??**以外のプログラム群は一切の変更を加えずに利用することができ，これがモジュール群に分割した実装の利点である。