分散分権型環境での機械学習とリスク管理

井手剛

2021 年 10 月 31 日

目　次

第 1 章
分散分権型環境での機械学習とリスク管理

異常検知モデルの実際の適用において常に問題となるのは、異常事例の数が一般に極めて少ないという点である。この点への実用的な対処策として、異なるデータ源、たとえば、複数の異なる会社で収集された分散データを統合しモデルを学習する、という方法が考えられる。そのようなシナリオにおいては、データプライバシーを保護しつつ学習を行うにはどうすべきか、データ源同士に分布のずれが存在しうる場合にどうモデルを構築すべきか、という 2 つの問題が生ずる。データの所有者が複数いるとすれば、全体の学習機構は中央集権的ではなく分権的であることが望ましい。この「分散分権」という設定においては、「分散台帳」とも称されるブロックチェーン技術との関係が興味深い。本章では、分散分権型の学習問題の定義を与えたのち（1.1 節）、その要請を満たす異常検知モデルの例を与える（1.2 節）。次いでビットコインの目指した世界を道しるべとしつつ（1.3 節）、合意形成の理論（1.4 節）とその具体的なモデル（1.5 節）について述べる。

1.1 分散分権型の学習問題

本章で興味を持つ**分散分権型** (decentralized) 機械学習の問題設定について、類似の問題との対比において簡単に見ておこう。

本章で考えるのは、図 1.1 のようにネットワークに接続された S 個のデータ生成源 $s = 1, \ldots, S$ があり、それぞれデータ集合 $\mathcal{D}^1, \ldots, \mathcal{D}^S$ を保持しているという状況である。各データ生成源のことを参加者 (member) と呼び、図のような S 人の参加者の集まりを系 (system) と呼ぶことにする。分散分権型学習の問題設定は

- 学習における計算は中央のサーバーを必要とせず参加者の手元で行うこと。
- 学習の結果、個々の参加者が個別の状況に応じて一般には異なるモデルを得ること。
- 参加者のデータやモデルは他の他の参加者には共有されないこと。

という 3 点を特徴とする。

第 1 の点は、いわば参加者に**民主主義** (democracy) を保証することである。全ての参加者は同格で、ネットワークを介して互いに通信することができる。社会基盤としてのネットワークおよびルーター、したがって共有されたクロックの存在は仮定されるが、例えば全員からデータを集めて確率的勾配法を回すようなサーバーは存在しない。

第 2 の点は、いわば参加者に**多様性** (diversity) を保証することである。図の設定であれば学習の結果は S 個の一般には異なるモデルである。「モデル」とは一般に、データに内在するパターンを表現した確率分布のことを指す。もしデータに入力ベクトル $\boldsymbol{x}$ とラベル y の対が保存されていれば、$\boldsymbol{x}$ を与えた時にラベル y の値を予測するための条件付き確率分布 $p(y \mid \boldsymbol{x})$ を求めることが目標になる。これは分類または回帰問題の解に対応する。一方、ラベル情報が与えられていない場合は、確率分布 $p(\boldsymbol{x})$ を求めることが問題になる。これは密度推定の問題であり、後で詳しく論ずる。

第 3 の点は、参加者に**プライバシー** (privacy) を保証することである。ここでは、話を単純化するため、悪意の参加者は存在せず、彼らは「正直だが好奇心にあふれる (honest but curious)」存在であると仮定する。すなわち、あえて偽のデータを使って系全体の学習を妨害したり、共謀してプライバシーを破ったり、といったことを行う参加者はいないものとする。これは企業間のコンソーシアムのような形態であれば十分現実的な設定である。

本書の主題である異常検知の場合、それぞれのデータ集合 $\mathcal{D}^1, \ldots, \mathcal{D}^S$ が正常状態において取得されたと仮定されることが多い。第 1 章で述べた通り、訓練データを使って正常状態のモデルを作り、新たな標本を観測した時に、典型的には異常度 $-\ln p$ がある値を上回れば（もしくは、確率値がある値を下回れば）異常と判定するわけである。確率分布の学習のためには正常標本だけあればよく、異常標本は必要ないが、異常検知モデルの構築のためには、異常度に適切なしきい値を付す必要がある。そしてそのためには、なるべく多くの異

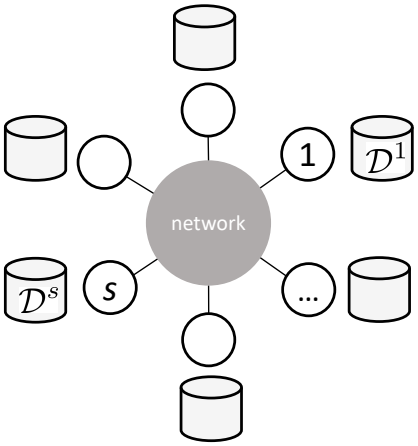

図 1.1　分散分権型の機械学習の場面設定

常事例を収集することが必要である。しかし異常の発生は一般に歩留まりなど重大なビジネス上の情報に直接関係し、機密扱いになることが普通である。異なる企業間ならもちろんのこと、同一の企業の支社間や事業所間ですら、データが完全に共有されるという想定は非現実的である。したがって参加者には、他人からの情報収集と、自分のデータの機密保持との間でトレードオフが存在する。

分散学習 (distributed learning) や**連合学習** federated learning) の場合、S 個のデータ集合を何らかの意味で連携させて単一のモデルを構築することを目標とするのが通例である。分散分権型の場合、やはり S 個のデータ集合を何らかの意味で連携させるが、今回の場合は中央のサーバーの存在を前提にすることはできないため、各データ生成源において別個のモデルを学習することになる。つまり学習の目標は単一のモデルを得ることではなく、S 個の一般には異なったモデルを得ることである。機械学習の用語ではこれは**マルチタスク学習** (multi-task learning) とも呼ばれる。言うまでもなくマルチタスク学習はシン

グルタスク学習を特別な場合として含む。この意味で、本章での設定は、通常の意味での分散学習・連合学習の一般化になっている。

1.2 多様性を保証するための異常検知モデル

前節で、分散分権型学習は、民主主義、多様性、プライバシーという 3 つ制約条件をもつ学習の問題であることを述べた。まず全体像をつかむため、民主主義制約とプライバシー制約がないとして、多様性を保証するためにはどういう学習方法が可能かを考えてみる。

ラベルなしのデータが与えられる教師なし密度推定の設定において、各参加者は次のようなデータ集合を持っているとする。

$$\mathcal{D}^s = \{\boldsymbol{x}^{s(1)}, \ldots, \boldsymbol{x}^{s(N^s)}\} \tag{1.1}$$

それぞれの測定値は M 個の実数値から成るとする。すなわち、s 番目の参加者は N^s 個の M 次元ベクトルを保持している。以下、第 s 番目の参加者の観測値を表す確率変数を $\boldsymbol{x}^s \in \mathbb{R}^M$ とし、その実現値を上記のように上付きのカッコをつけて区別する。たとえば $\boldsymbol{x}^{s(n)}$ は、s 番目の参加者の持つデータ集合における第 n 番目の標本である。また、混乱がない限り、$p(\cdot)$ という記号を一般に確率密度関数を表す記号として使う。たとえば $p(\boldsymbol{\theta}_k)$ と $p(\boldsymbol{\pi}^s)$ は異なる分布を表しているので注意されたい。学習の目標は、個々の参加者の観測値について $\boldsymbol{x}^s$ が従う確率分布を求めることである。

なお、以下に述べる共同辞書学習では、ノイズを含むセンサーの測定値のようなデータに内在する複雑性が高いような状況を想定している。これは性別や疾病履歴のような単純で、それ自体を「読める」データとは異なり、データのすべてのビットを厳密に管理することよりも、全体の傾向、すなわち確率分布を学習することの方にむしろ興味があるということである。人間が情報を「読める」のは確率分布という集合体からであって、個々の標本の個々の桁の数値ではない。確率分布を正確に学習することは機械学習では教師なし学習と言われ、一般に難しい。難しいからこそ、他の参加者と協業する必要がある。

1.2.1 確率モデルの設定

多様性を保証するための最も単純な方法は、各参加者がまったく別のモデル

に従うとするものである。しかしこれではそもそも共同して学習を行う意味がない。そこでここでは、S 人の参加者が K 個の確率モデルを共有していると想定し、以下のような混合モデルを考える。

$$p(\boldsymbol{x}^s \mid \boldsymbol{\Theta}, \boldsymbol{z}^s) = \prod_{k=1}^{K} f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k)^{z_k^s} \tag{1.2}$$

$$p(\boldsymbol{z}^s \mid \boldsymbol{\pi}^s) = \prod_{k=1}^{K} (\pi_k^s)^{z_k^s}, \tag{1.3}$$

ここで、$\boldsymbol{z}^s \in \{0,1\}^K$ は「s 番目のモデルが K 個の要素のどれから来たのか」を表現する指示ベクトルで、 $\sum_{k=1}^K z_k^s = 1$ のようにいわゆる one-hot 式に定義される。$\mathrm{Cat}(\cdot \mid \boldsymbol{\pi}^s)$ は確率値 $\pi_1^s, \ldots, \pi_K^s$ をパラメターとするカテゴリカル分布を表す。もちろん $\sum_{k=1}^K \pi_k^s = 1$ を表す。$\boldsymbol{\Theta} \triangleq \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K\}$ は K 個の要素のモデルパラメターをまとめたものである。なぜこれが混合モデルを表しているかは、$\boldsymbol{z}^s$ でモデルを周辺化してみるとわかる。すなわち

$$\sum_{\boldsymbol{z}^s} p(\boldsymbol{x}^s \mid \boldsymbol{\Theta}, \boldsymbol{z}^s) p(\boldsymbol{z}^s \mid \boldsymbol{\pi}^s) = \sum_{k=1}^{K} \pi_k^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k) \tag{1.4}$$

が成り立つ。ただし $\boldsymbol{z}^s$ での和というのは、$(1,0,\ldots,0)^\top, \ldots, (0,\ldots,0,1)^\top$ のような K 種類の指示ベクトルにわたる。もしも $K = 3$ なら $(1,0,0)^\top, (0,1,0)^\top, (0,0,1)^\top$ という 3 つである。この式から π_k^s が、「参加者 s のモデルにおいて、第 k 番目のデータ生成パターンが採用される確率」という意味を持っていることが分かる。たとえば、ひとつのデータ生成パターンしか持たないような場合、ひとつの k において π_k^s の値が 1 になり、他は 0 である。このモデルにおいては、混合重み $\boldsymbol{\pi}^s$ が s に依存しているという事実が多様性を表現するカギになっている。異なる s ごとに混合重みが違うので、式 (1.4) は、確率変数 $\boldsymbol{x}^s$ についてのそれぞれ異なる分布になる。それぞれの s において、訓練データ $\mathcal{D}^s$ に必ずしも含まれない新しい観測値を得たとすると、この分布に照らして確率が高いか低いかで正常か異常かを判断できることになる。これを可能にするためにはパラメター $\{\boldsymbol{\pi}^s\}_{s=1}^S, \{\boldsymbol{\theta}_k\}_{k=1}^K$ をデータから定めなければならない。

話を一般的にするため、$\boldsymbol{\Theta} \triangleq \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K\}$ は

$$p(\boldsymbol{\Theta}) = \prod_{k=1}^{K} p(\boldsymbol{\theta}_k) \tag{1.5}$$

という事前分布を持つと考える。また、混合重み $\boldsymbol{\Pi} \triangleq \{\boldsymbol{\pi}^1, \ldots, \boldsymbol{\pi}^S\}$ については、ディリクレ分布を事前分布として設定する。

$$p(\boldsymbol{\Pi} \mid \gamma) = \prod_{s=1}^{S} p(\boldsymbol{\pi}^s) = \prod_{s=1}^{S} \frac{\Gamma(K\gamma)}{\Gamma(\gamma)^K} (\pi_1^s \cdots \pi_K^s)^{\gamma-1} \tag{1.6}$$

これらの事前分布はモデルに特段の偏見を導入するものというよりは、数値計算を円滑にし解の性質を改善するためのものである。あとで見るとおり、$p(\boldsymbol{\theta}_k)$ は M 次元の測定値の間の非本質的な関係を捨象するのを助け、$p(\boldsymbol{\pi}^s)$ は K 個の要素の間の不釣り合いに起因する数値的不安定性を除去する役割を持つ。

1.2.2　対数尤度の式と $\{\boldsymbol{\pi}^s\}$ についての解

さて、観測モデル (1.2) とそれに付随する事前分布を式 (1.3) と式 (1.5)-(1.6) において設定したので、最大事後確率 (maximum a posteriori; MAP) 推定の枠組みで点推定を行うことを考えよう。要素割り当ての指示ベクトル $\mathsf{Z} \triangleq \{\boldsymbol{z}^{s(n)}\}$ は非観測変数なので、MAP 推定においてはこれを周辺化したものを最大化することになる。対数尤度は結局

$$L_0(\boldsymbol{\Pi}, \boldsymbol{\Theta}) \triangleq \ln \left\{ p(\boldsymbol{\Pi}) p(\boldsymbol{\Theta}) \sum_{\mathsf{Z}} \left[\prod_{n,s} p(\boldsymbol{x}^{s(n)} | \boldsymbol{\Theta}, \boldsymbol{z}^{s(n)}) p(\boldsymbol{z}^{s(n)} | \boldsymbol{\pi}^s) \right] \right\}, \tag{1.7}$$

のようになる。

この対数尤度は、s という参加者を区別する添え字がある以外は通常の混合モデルと同じである。混合モデルに対する通常の処方箋に従い、Jensen の不等式を用いて L_0 の下限 L を導き、その下限を最大化することでパラメター推定を行うことを考える。Jensen の不等式を Z の和に対して適用することで

$$\begin{aligned} L_0(\boldsymbol{\Pi}, \boldsymbol{\Theta}) \geq L(\boldsymbol{\Pi}, \boldsymbol{\Theta}) &\triangleq \mathrm{c.} + \ln[p(\boldsymbol{\Pi}) p(\boldsymbol{\Theta})] \\ &+ \sum_{\mathsf{Z}} Q(\mathsf{Z}) \sum_{s,n} \ln[p(\boldsymbol{x}^{s(n)} | \boldsymbol{\Theta}, \boldsymbol{z}^{s(n)}) p(\boldsymbol{z}^{s(n)} | \boldsymbol{\pi}^s)], \end{aligned} \tag{1.8}$$

を得る。ただし c. はパラメターに依存しない定数で、Jensen の不等式により導入された新しい分布を $Q(\mathsf{Z})$ と書いた。

MAP 推定を完全に実行するためにはもちろん $\{f(\boldsymbol{x}^s|\boldsymbol{\theta}_k)\}$ と $\{p(\boldsymbol{\theta}_k)\}$ の具体的な関数形を与えなければならない。しかし $Q(\mathsf{Z})$ と $\{\boldsymbol{\pi}^s\}$ についてはこの時点で形式的な解を求めることができる。まず $Q(\mathsf{Z})$ について考えよう。対数周辺化尤度 (1.8) の式の形を見ると、s, n について和の形になっており、各 (s, n) について独立に考えることができる。そこで $Q(\mathsf{Z}) = \prod_{s=1}^{S}\prod_{n=1}^{N^s} q(\boldsymbol{z}^{s(n)})$ とおき、L を最大化するように $q(\boldsymbol{z}^{s(n)})$ を決める。この前提で尤度の下限を書き直してみると

$$
\begin{aligned}
L(\boldsymbol{\Pi}, \boldsymbol{\Theta}) = \text{c.} + \ln p(\boldsymbol{\Theta}) + \sum_{s=1}^{S}\sum_{k=1}^{K}(\gamma - 1)\ln \pi_k^s \\
+ \sum_{s=1}^{S}\sum_{n=1}^{N^s}\sum_{k=1}^{K}\sum_{\boldsymbol{z}^{s(n)}} q(\boldsymbol{z}^{s(n)}) z_k^s \ln\left[\pi_k^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k)\right] \qquad (1.9)
\end{aligned}
$$

のようになる。これを最大化する $q(\boldsymbol{z}^{s(n)})$ を、規格化条件を満たすように定めたい。これは $q(\boldsymbol{z}^{s(n)})$ という関数にわたる最適化問題になるので難しく聞こえるが、指示ベクトル $\boldsymbol{z}^{s(n)}$ は K 通りの値しかとらないので、要するに K 個の確率値を定める問題である。ある特定の (s, n) に着目して、$K = 3$ で考えてみよう。$k = 1, 2, 3$ の時の確率をそれぞれ $r_1^{s(n)}, r_2^{s(n)}, r_3^{s(n)}$ とすると、$L(\boldsymbol{\Pi}, \boldsymbol{\Theta})$ において関係する部分は

$$
r_1^{s(n)} \ln\left[\pi_1^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_1)\right] + r_2^{s(n)} \ln\left[\pi_2^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_2)\right] + r_3^{s(n)} \ln\left[\pi_3^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_3)\right]
$$

となる。規格化条件 $\sum_{l=1}^{3} r_l^{s(n)} = 1$ をラグランジュ係数 λ を使って取り込むと、最適条件は

$$
\frac{\partial}{\partial r_k^{s(n)}}\left[\sum_{l=1}^{3} r_l^{s(n)} \ln\left[\pi_l^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_l)\right] - \lambda \sum_{l=1}^{3} r_l^{s(n)}\right] = 0
$$

となる。これを解いて $r_k^{s(n)} \propto \pi_k^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k)$ が得られ、通常の混合モデルとまったく同様の

$$
Q(\mathsf{Z}) = \prod_{s=1}^{S}\prod_{n=1}^{N^s}\prod_{k=1}^{K}(r_k^{s(n)})^{z_k^{s(n)}} \qquad (1.10)
$$

$$r_k^{s(n)} = \frac{\pi_k^s f(\boldsymbol{x}^{s(n)} \mid \boldsymbol{\theta}_k)}{\sum_{m=1}^K \pi_m^s f(\boldsymbol{x}^{s(n)} \mid \boldsymbol{\theta}_m)} \tag{1.11}$$

のような解が得られる。ラグランジュ係数は規格化条件 $\sum_{k=1}^K r^{s(n)} = 1$ により消去できることに注意。さらに、$\boldsymbol{\pi}^s$ についても、規格化条件をラグランジュ係数で取り込みつつ、L を π_k^s について微分して 0 と置くことで

$$\pi_k^s = \frac{N_k^s + \gamma - 1}{N^s + K(\gamma - 1)} \tag{1.12}$$

となることが簡単にわかる。ただし $\gamma - 1$ は 1 のオーダーの定数で、固定と考える。また

$$N_k^s \triangleq \sum_{n=1}^{N^s} r_k^{s(n)} \tag{1.13}$$

と定義した。N^s は参加者 s が持っている標本の総数で、N_k^s はそのうちパターン k に属すると思われる標本の数という解釈ができる。先に述べたように、特定のパターンを好むような参加者の場合、単一の k においてのみ $N_k^s > 1$ となることもあるかもしれない。その場合、他の k で π_k^s はほぼ 0 にならざるを得ないが、目的関数 (1.9) は $\ln \pi_k^s$ という項を含むため、0 は数値的不安定性を引き起こす。先に設定したディリクレ事前分布は、$\gamma > 1$ を与えることでそのような不安定性を除去する役割を果たす。

明らかに式 (1.11)-(1.12) は未知パラメター $\{\boldsymbol{\theta}_k\}$ に依存している。したがって、モデル推定のためには、最初に何らかの方法でパラメターを初期化し、$Q(\mathsf{Z})$ の計算と $\{\boldsymbol{\theta}_k\}$ の推定を交互に行う。すなわち、最初に $\{\boldsymbol{r}^{s(n)}\}$ を初期化し、$Q(\mathsf{Z})$ が分かっている前提で L を最大化することで $\boldsymbol{\Pi}, \boldsymbol{\Theta}$ を求める。次に今求めたパラメターを前提にして、$Q(\mathsf{Z})$ が再評価される。これの枠組みは通常の混合モデルの EM 法とまったく同一である。違いは、マルチタスク学習となっているため、$s = 1, \ldots, S$ について異なった混合モデルができることである。

1.2.3　指数型分布族に対する一般解

パラメター $\{\boldsymbol{\theta}_k\}$ の分散環境下での学習に関して、観測モデルがいわゆる指

数型分布族

$$f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k) = G(\boldsymbol{\theta}_k)H(\boldsymbol{x}^s)\exp\left\{\boldsymbol{\eta}(\boldsymbol{\theta}_k)^\top \boldsymbol{T}(\boldsymbol{x}^s)\right\}, \tag{1.14}$$

に従うと仮定してみる。ここで $H(\cdot), G(\cdot)$ はあるスカラー関数で、$\boldsymbol{\eta}(\cdot), \boldsymbol{T}(\cdot)$ は列ベクトルを返すベクトル値の関数で、規格化条件を満たすように決められる。指数型分布族の代表的な例はガウス分布やガンマ分布が挙げられる。その選択には任意性があるが、合意形成を行うためには、$s = 1, \ldots, S$ にわたる全参加者が同一の分布を使うことが必要である。この「何について合意するかを合意する」という問題はしばしば**メタ合意形成** (meta consensus) 問題と呼ばれ、実応用上の大きな問題になりうる。

このような表現を尤度の下限の式 L (式 (1.8)) に入れ、$\{\boldsymbol{\theta}_k\}$ に関係する部分のみを拾うと

$$\sum_{k=1}^{K}\left[\ln p(\boldsymbol{\theta}_k) + \sum_{s=1}^{S}\left\{N_k^s \ln G(\boldsymbol{\theta}_k) + \boldsymbol{T}_k^{s\top}\boldsymbol{\eta}(\boldsymbol{\theta}_k)\right\}\right], \tag{1.15}$$

のようになる。ただし $\boldsymbol{T}_k^s \triangleq \sum_{n=1}^{N^s} r_k^{s(n)} \boldsymbol{T}(\boldsymbol{x}^{s(n)})$ と定義した。この表式から分かる通り、もし

$$N_k = \sum_{s=1}^{S} N_k^s, \qquad \boldsymbol{T}_k = \sum_{s=1}^{S} \boldsymbol{T}_k^s, \tag{1.16}$$

という量が計算できさえすれば、パラメター $\{\boldsymbol{\theta}_k\}$ は式 (1.15) を最大化することで求められる。すなわち

$$\boldsymbol{\theta}_k = \arg\max_{\boldsymbol{\theta}_k}\left\{\ln p(\boldsymbol{\theta}_k) + N_k \ln G(\boldsymbol{\theta}_k) + \boldsymbol{T}_k^\top \boldsymbol{\eta}(\boldsymbol{\theta}_k)\right\} \tag{1.17}$$

である。これを解くには $G(\cdot), \boldsymbol{T}(\cdot), \boldsymbol{\eta}(\cdot)$ の関数形を具体的に与えなければならないが、それをせずとも、指数型分布族に対しては、EM 反復の全体が非常に興味深い構造を持っていることが見て取れる。すなわち、反復の 1 回が、各参加者の手元での局所的更新、すべての参加者にわたる合意形成、そして最適化、という 3 つのステップに分離されている。このモデルでは、S 人の参加者が、K 個のパターンの集まりを共同で学習する。パターンの集まりをいわば辞書のように見て、辞書から自分の個別の状況に適合するパターンを適

切な重み $\boldsymbol{\pi}^s$ をつけて混合する。この意味でこの枠組みを**共同辞書学習** (collaborative dictionary learning) または**共同パターン辞書学習** (collaborative pattern dictionary learning) と呼んでよい[6], [10]。算法の大枠を Algorithm 1 にまとめておく。

Algorithm 1 共同パターン辞書学習

1: パターン辞書 $\{\boldsymbol{\theta}_k\}_{k=1}^K$ を初期化する。
2: 各参加者 s において $\{r_k^{s(n)}\}$ を初期化する。
3: **repeat**
4:　　局所的更新: 現在のパターン辞書を使い、各参加者 s において、自分のデータの標本重み $\{\boldsymbol{r}^{s(1)}, \ldots, \boldsymbol{r}^{s(N^s)}\}$ と、局所的総和 $\{\boldsymbol{T}_1^s, \ldots, \boldsymbol{T}_K^s\}$ および $\{N_1^s, \ldots, N_K^s\}$ を計算する。
5:　　合意形成: 式 (1.16) をプライバシー制約の下計算する。
6:　　最適化: 式 (1.17) を解いて各参加者がパターン辞書を更新し保存する。
7: **until** 収束

この算法においては、各参加者は、自分の手元にあるモデルを反映させた最適モデルを手に入れるというインセンティブが存在する。つまり、偽のデータを使って他の参加者の情報を得ようとすると、共同学習される辞書の品質が必然的に下がる。なぜなら偽のデータに対して最尤なモデルが得られてしまうからである。分布の裾の部分まである程度自信を持ったモデルを構築するためには、文字通り最尤推定の教えるとおりに行動するしかない。言い換えると、尤度がこの共同学習の参加者の定量的インセンティブとなっている。

1.3　分権型合意形成問題

前節に導いた指数型分布族に対する共同辞書学習問題において、分散分権型学習の観点で特に興味深いのは合意形成の部分である。それ以外の部分は参加者の手元で局所的に計算が済むが、合意形成の部分は他者との通信が必ず必要である。上記の EM 算法における合意形成の問題とは、ひと言でまとめれば秘匿集計である。各参加者 s は、式 (1.16) における総和を、N_k^s や $\boldsymbol{T}_k^s$ を、他の参加者 $s' \neq s$ に開示することなく計算したい。分権型でない場合はこれは中央のサーバーとの間に安全な通信手段を確保できれば済む話なので簡単であるが、分散分権型の設定ではこれは困難な問題になる。

これまで論じてきた分散分権型の密度推定の問題とは少し離れるが、分権型合意形成の方法を考える上では、ブロックチェーン技術は避けて通れない。この点について以下少し考えてみよう。

1.3.1 サトシ・ナカモトの挑戦

分散分権型の環境におけるセキュリティリスクを考える上での今日的話題として、いわゆる**ブロックチェーン** (blockchain) 技術との関係は興味深いところである。今日ブロックチェーン技術と呼ばれるものは、ビットコインの基盤技術を提案した Satoshi Nakamoto なる素性不明の人物のプレプリント論文[12]に源を発する[1]。Nakamoto の目的のひとつは、政府から通貨発行権という強大な権力を奪うことによる社会の究極の民主化であったと言われている。通貨の本質とは何か。現代の国家で発行されている紙幣は文字通り紙切れでありそれにモノとしての価値は乏しい。だとすればその価値の本質は、「それがある特定の価値を持っているという合意」そのものにある。国家が発行する紙幣は、その合意の証書として作用する。いわゆる電子マネーの世界に行っても、電子マネーの 1 万円は常に同額の紙幣と交換可能であるという意味において、その合意は国家権力により強制されたもの、とみなすことができる。そのためしばしば国の発行する通貨は fiat currency などと言われる。Fiat は通常の単語としては独善とか専断という意味である。

しかしながら、通貨が「合意された価値」を表すものとすれば、その本質は合意そのものにあり、国家による強制は必ずしも必要ではない。国家がそこに介在しているのは、議会民主制や世襲の王権授受を通して、何となく「皆の代表」という雰囲気を帯びており、なおかつ、警察力や軍事力といった強制手段を持っているという理由による。したがって法定通貨を認めることは、いかなる国家権力でも多かれ少なかれ帯びる独裁的傲慢の前に膝を屈することでもある。これは正しいことなのか、というのがひとつの目的意識である。逆に言えば、何らかの技術により、誰が見ても公平な価値合意の認証作業ができれば、国家の強制によらない「正しい」通貨を構築できるのではないだろうか。国家を介在させず、したがって国境の壁も超えて、地球上の誰しもがその価値につ

[1] プレプリントとは、いわゆるピアレビュー（同じ分野の研究者による査読）を経ていない論文のこと。

いて合意を形成できる仕組みを作ることができれば、それは文字通りの世界革命、ジョン・レノンが名曲『イマジン』で夢見たような、国家権力からの人類の解放である。

Nakamoto 論文には、大きく分けて二つの内容が書かれている。ひとつは、暗号通貨の発生から個々の取引までの来歴を、本人認証をしつつ記録する仕組みである。これは電子署名の技術と、送金等の商取引のたびに数珠つなぎに取引情報のハッシュ値をつないでゆくデータ構造（ハッシュチェーン）を主な特徴とする。ただし、ハッシュチェーンそれ自体は昔から普通に知られていた概念で、取り立ててそこに技術革新があるわけではない[24]。

もうひとつは、それらの取引記録を検証し承認する仕組み、すなわち合意形成の仕組みであり、ここがブロックチェーン技術の本質と言えよう。ビットコインの文脈では、これは通貨の 2 重使用を防ぐ仕組みとして働く。たとえば、A 氏が B 氏に何かの対価として 10 ビットコイン（BTC）を送る場合、A 氏の電子通帳には −10 BTC、B 氏の通帳には +10 BTC というやり取りが永久に記録されなければならない。そしてこの情報は暗号通貨を保持する人全員に共有されねばならない。さもなくば、その送金履歴を消去した上で、A 氏は別の C 氏に 10 BTC を送金して、たとえば商品を詐取することができてしまうからである。ビットコインの実際の運用では、およそ 10 分間ごとに未承認の全取引記録をひとかたまりにまとめ（それをブロックと呼ぶ）、そのブロックの検証・承認が「**採掘者** (miner)」と呼ばれる人たちにより行われることになっている。採掘者が検証を済ませるまで、取引にかかわった人は待たされる。もし悪意をもつ参加者がいれば、いったん使った通貨を何度も使い、それを正当な取引記録として承認しようとするだろう。これを防ぐために工学的に有効な解決策を提示した、というのがブロックチェーンの革新的なところである。

1.3.2　ビザンチン将軍問題

合意形成の問題は計算機科学で古くから研究されてきた問題であり、たとえばビザンチン将軍問題はその典型例としてよく知られている。最も単純な設定では、問題は次の通りである。S 人の参加者（将軍）のそれぞれが、他の $S-1$ 人と個別に会話し、ある他国を攻撃するかしないかという 2 値のどちらかを表明する。全員が全員と話し終わった時点で、攻撃するか攻撃しないかのどちら

かを多数決で決めたい。攻撃するとしたら全員で行かないと返り討ちに会うかもしれないので、S 人全員での合意を形成したい。これのどこが難しいのか、と思われるかもしれないが、2 者間通信しか許されていないこと、二枚舌、三枚舌の将軍がいるかもしれないこと、という 2 点を考えると話が一気に難しくなる。二枚舌というのは、たとえば将軍 1 が、将軍 2 に伝えた意思表明と、将軍 3 に伝えた意思表明が違う、というような場合である。そのような場合、繰り返し 2 者間の会話を行い、手元にある最新の多数決の結果を伝え合うことで、要するに誰が信頼できないのかあぶり出すことができる。典型的には、信頼できない参加者が $\frac{1}{3}S$ 人未満だとなんとか「正しい」多数決に到達できると言われている。しかしそのためには指数関数的な通信回数が必要になり、少なくとも暗号通貨のような大衆的な応用では、少なくともそのままの形では技術的に実行困難であると考えられている。

ビットコインにおけるブロックの承認問題はビザンチン将軍問題と似ている。S 人の採掘者が、承認か非承認かを多数決で決めれば、悪意の人が 3 割未満である限り合意を形成できそうに見える。実際、参加者が限定されたプライベート型のブロックチェーンでは、そのような仕組みが実装されることがある。そういう背景もあり、ブロックチェーンが、ビザンチン将軍問題という合意形成問題への画期的な解を与えたという解説（たとえば[25], [27]）も多く見るが、正直、意味がよくわからない。Nakamoto 論文[12] ではビザンチン将軍問題への言及はまったくないし、当然、そのような証明は書かれていない。通常の意味でビザンチン将軍問題の解決を証明できる可能性もおそらくない。ビットコインの意思決定は多数決ではなく、約 10 分間の取引記録をまとめた「ブロック」という単位ごとに、ただ一人の承認者がブロックの正しさを決める。ビザンチン将軍問題は比喩としてはよいとしても、少なくとも Nakamoto 論文で提案されたビットコインの動作に関する限り、数学的には無関係と思った方がよい。伝統的な合意形成手法と、次節で述べるビットコインの確率的性格の対比については、たとえば最近の総説論文[19] が参考になろう。

1.3.3　「労力の証明」：ビットコインの合意形成手法

Nakamoto 論文の技術革新はむしろ、ビザンチン将軍問題に代表されるような伝統的な合意形成の方法とまったく異なった方法で、取引データの正当性に

ついての合意を形成するという点にある。すなわち、ビットコインにおいては、実はブロックの承認過程に予測不可能性を入れることで安全性を持たせている。この発想は、商用の金融システムの設計思想としては極めて斬新である。ビットコイン以前のほとんどすべての金融システムは、システムのあらゆる動作から予測不可能性、曖昧性を排除する方向で設計されてきた。たとえばビットコインでも採用されている電子署名システムでは、秘密鍵と公開鍵が代数的に双子の関係にあることを利用して、代数方程式の解として曖昧性なく本人確認ができるようになっている。システムが、本人かもしれないし、そうでないかもしれない、というような回答をすることはあり得ない。

ビットコインでは、下記に示す通りある意味で多くの採掘者の中からランダムに一人を選び、直近の取引記録の詰まったブロックを承認する権利を与える。それが「**労力の証明** (proof-of-work)」と呼ばれる有名な仕組みである。選ばれた人は取引記録のハッシュチェーンをたどり矛盾がないかを確認し、新たに検証したブロックを、検証済みのブロックの先頭に加える。これが「ブロックチェーン」という用語の由来である。承認権限を得られるかどうかは予測不能で、確率的にしかわからない。悪意を持った採掘者が「当たり」を引くかもしれないし、そうでないかもしれない。しかし、後述のように、悪意を持った採掘者が少数派である限りにおいて中長期的には取引記録は無矛盾なものへと収束することが期待される。ここにも社会の究極の民主化を目指した彼らの理想主義が垣間見えて興味深い。国家や金融機関というある種の独裁権力に我々の判断を預けることは不正義だと彼らは考える。我々の中に悪意を持った者が一定数いるという現実を受け入れつつ、仮に一時的に問題が生じても、速やかに正しい状態に戻せるような分権的な仕組みはどういうものなのか。それは数学の証明問題というよりは、社会的制度設計の問題である。

具体的には承認者の選択は次のように行う。各採掘者は、最後に承認されたブロックのハッシュ値 b_0（何かの整数）と、今承認しようとしているブロックに含まれる取引記録をひとまとめにして整数で表現したもの c を入力として、新しいハッシュ値 b を次のように計算する

$$b = h(b_0, c \mid \kappa) \tag{1.18}$$

ただし h は所定のハッシュ関数、κ は調節可能なパラメター（整数）である。

ざっくり言えば、ここに出てくる 3 つの数字をつなげてひとつの数字を作り、そのハッシュ値を求めるということである。ハッシュ関数の性質からして、κ を少しでも変えると、結果として得られるハッシュ値 h はまったく異なるものになる。承認者となるためには、出てきたハッシュ値があらかじめ決められた整数より小さくなるような κ を見つけなければならない。この「あらかじめ決められた整数」というのは、承認の時間間隔がうまく想定と一致するようにシステム管理者により決められる。結果が予想できないがゆえ、総当たり的に試すしかない。幸運にもうまい κ を見つけた採掘者は、それを手に承認者として名乗りを上げるわけである。

ここで極めて重要なことは、ハッシュ関数の値は予測不能ということである。ハッシュ関数がまともなものならば、たとえば $\kappa = 1000$ の時のハッシュ値と、$\kappa = 1001$ の時のハッシュ値には何の相関もない。だから、これまで試した計算結果から、よさそうな κ の値を逆算するというようなことは不可能である。そのため、労力の証明の過程には乱数はどこにも入っていないにもかかわらず、実質的に、承認者は候補者のなかからランダムに選ばれることになる。例えば、採掘者全員が似たような計算機能力を持っていたとしよう。だれかが当たりを引くまでに試せる κ の個数は皆似たようなものだから、ハッシュ関数の予測不可能性に照らせば、実質的には承認者にわたる一様分布から、承認者を無作為抽出しているのと同じである。一般には計算機能力にはばらつきがあるので、採掘者 m の計算機の能力を CPU_m とすると

$$p_m = \frac{\mathrm{CPU}_m}{\sum_{m'=1}^{N_{\mathrm{miner}}} \mathrm{CPU}_{m'}} \tag{1.19}$$

という確率を、N_{miner} 人の採掘者にそれぞれ割り振り、$p_1, \ldots, p_{N_{\mathrm{miner}}}$ をパラメターとするカテゴリカル分布から標本抽出しているのと同じである。確率分布からの標本抽出であるがゆえ、最大の計算機能力を持つ者が常に選ばれるとは限らない。したがってビットコインにおける承認者の選択は計算機能力による多数決とは言えない。取引を確定させるための実際の行動が確率的にしか決められないという性質を、しばしば**確率的決着性** (probabilistic finality) と呼ぶ。

選択がランダムなのであれば、そもそもハッシュ値の総当たり的な計算など無駄ではないか、と考える人もいると思う。ある意味それは正しい。これから

先は数学というよりむしろ心理学の世界である。ビットコインでは採掘に成功するたびに所定の賞金が支払われることになっている。それをインセンティブにして採掘者は計算機資源に多大な投資を行う。彼らが、自分の儲け口である採掘という仕事を台無しにするような行為をする可能性は低い。そもそもなぜ 2 重送金をしたいかといえば、楽をしてお金を手にしたいからである。そういう不逞の輩が、わざわざ手間暇かけて採掘競争に参入するだろうか、というのが裏にある論理である。いわば計算機の能力を「善人度」の指標としているわけである。

また、逆に言えば、選択がランダムなのであれば、悪意の参加者をたまたま選んでしまう確率もあるではないか、という非難も成り立ちうる。それも正しい。ビットコインの取引記録の妥当性の保証は確率的にしか与えられない。口座残高が 100% 確実に正しいという前提に立つ通常の（中央集権的な）取引とは発想が根本的に異なる。ではその確率とやらはどの程度なのか。この疑問に答えるため、Nakamoto 論文では最後に、簡単な確率モデルを使ってその見積もりをしている。その内容を次節で見てみよう。

1.3.4　ギャンブラーの破産問題

前節でひとつのブロックの承認過程について説明した。新たに承認者に指名された採掘者は、ブロック内部の取引の整合性を確認する。大量の取引記録の確認を一体どう行うのかと思われるが、ハッシュ木というデータ構造を使うと、整合性が取れているか否か（残高と送金記録が矛盾ないか、など）の質問へのイエス・ノーは高速に答えることができる。電子署名の技術により本人確認は確実に可能で、取引をハッシュチェーンに連ねることで取引の順序を確認することができても、一部の取引データを「握りつぶす」ことは可能である[23]。したがってもし承認者が悪意を持っていれば、自分の送金記録を改変し、2 重送金を行うことが可能である。

ビットコインの運営モデルでは、最も長いチェーン、すなわち、付加されたブロック数が最も多く、それゆえ、最も多くの採掘者により承認を受けたチェーンを正当な取引記録として参照することになっている。それゆえ、ひとつの現実的な攻撃のシナリオとして、遠くない過去に行った自分の支払いを帳消しにするために、その取引を含むブロックから始めて最新のブロックまでのチェー

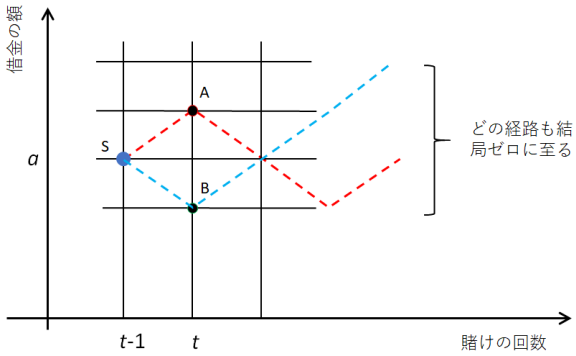

図 1.2　ギャンブラーの破産問題（借金帳消し問題）の説明

ンを自分で作り直す、という戦略が考えられる。話を単純にするために、採掘者を悪人軍団と善人軍団のふたつのグループに分ける。式 (1.19) で与えた確率から

$$p \triangleq \sum_{m\in\text{good}} p_m, \qquad q \triangleq \sum_{m\in\text{bad}} p_m \tag{1.20}$$

という二つの確率を定義する。p は善人軍団の中の誰かが承認権限を得る確率で、q は悪人軍団の中の誰かが承認権限を得る確率である。当然、規格化条件 $p+q=1$ が成り立つ。採用されるプロトコル（承認手順の規約）にも依存するが、最新の取引記録がブロックとして通知されるたびに、承認獲得競争に両者が参加し、もし善人軍団の誰かが勝てば正当なチェーンにブロックを追加し、悪人軍団の誰かが勝てば不当な取引を含むチェーンの方にブロックを追加する、という状況を想定する。このルールの下、悪人軍団は、z 個の正当なブロックを巻き戻したいとする。これを、最初に借金 z を持っていたギャンブラーと読み替える。ギャンブラーは、勝つ確率 q、負ける確率 p の賭けを続けて、借金をなんとか帳消しにしたい。借金が帳消しになった時点で、悪人軍団のチェーンが正当なものととして以降参照されることになる。

借金から出発しているので若干話がややこしいが、これは初等的な確率過程

論でよく知られた「**ギャンブラーの破産** (Gambler's ruin) 問題」と同じである。その名の示す通り、もともとこの問題は、「ある所持金から出発して賭けを続ける時、結局破産してしまう確率はいくらか」を問う。今の場合、「時刻 $t = 0$ において借金 z から始めて、任意の賭け試行の後に借金を帳消しにできる確率」を知りたい。その確率を R_z^0 とおこう。この問題が難しいのは、賭けの回数に制限がないことである。そのため、借金が帳消しになるという事象を数え上げることは困難である。そこで次のように考える。今、$t-1$ 回目の賭けを行う際に借金残高が a だったとする。この状態（S とする）から出発して借金を帳消しにできる確率を R_a^{t-1} と書く。賭けの結果に応じて、借金の額はたとえば図に示したようなさまざまな経路をたどる。しかし S から出発した経路は必ず A または B のどちらかを通る。仮定により、A は確率 p（負けたので借金が増える）、B は確率 q（勝ったので借金が減る）で生ずるので

$$R_a^{t-1} = pR_{a+1}^t + qR_{a-1}^t \tag{1.21}$$

が成り立つ。ここで恒等式 $R_a^t = (p+q)R_a^t$ を辺々引いて整理すると

$$R_a^t - R_a^{t-1} = -(R_{a+1}^t - R_a^t)p + (R_a^t - R_{a-1}^t)q \tag{1.22}$$

が成り立つ。賭けの回数に制限がないことから、借金が帳消しになる確率は出発点の（横方向の）位置によらないはずなので、左辺はゼロのはずである。ゆえ

$$R_{a+1}^t - R_a^t = \frac{q}{p}(R_a^t - R_{a-1}^t) \tag{1.23}$$

である。この式は a がひとつ増えるたびに q/p が掛けられてゆくという形になっている。$a = 0$ なら開始時点ですでに借金がないので $R_0^t = 1$ であることに注意してこの式を書き下すと

$$R_2^t - R_1^t = \left(\frac{q}{p}\right)(R_1^t - 1)$$
$$R_3^t - R_2^t = \left(\frac{q}{p}\right)^2(R_1^t - 1)$$
$$\cdots$$

$$R_z^t - R_{z-1}^t = \left(\frac{q}{p}\right)^{z-1} (R_1^t - 1)$$

となる。辺々加えると

$$R_z^t - R_1^t = (R_1^t - 1) \sum_{j=1}^{z-1} \left(\frac{q}{p}\right)^j = (R_1^t - 1) \frac{\frac{q}{p} - \left(\frac{q}{p}\right)^z}{1 - \frac{q}{p}} \tag{1.24}$$

という結果が得られる。R_1^t を決めるためにはもうひとつの境界条件を使う。もし借金が無限大であれば（巻き戻すべき正当なチェーンの長さが無限であれば）借金を帳消しにできる可能性はないので、$R_\infty^t = 0$ のはずである。上式にこれを使うと、最終的に

$$R_1^0 = \begin{cases} 1, & p \le q \\ \frac{q}{p}, & p > q \end{cases}, \qquad R_z^0 = \begin{cases} 1, & p \le q \\ \left(\frac{q}{p}\right)^z, & p > q \end{cases} \tag{1.25}$$

という結果が得られる。ただし、結果が開始点によらないはずであるということから、上付きの t を 0 に置き換えた。

この結果から、もし万が一、悪人軍団が承認権限を得る確率 q が善人軍団のそれ p よりも少しでも高いと、必ず借金を帳消しにできる、すなわち、任意のブロックにある取引を改変することができる。これがいわゆる「**51% 攻撃** (51% attack)」と呼ばれるものである。すなわち、式 (1.19) および式 (1.20) において、計算機の能力において善人軍団を上回ることができれば、承認の無作為選択による安全性を破ることができる。51% 攻撃以外にも、たとえば、採鉱に成功したタイミングを恣意的に動かせるという自由度を利用した Selfish mining という攻撃[3] など、いくつかのセキュリティリスクが知られている。

そもそもビットコインの安全性は、無作為選択という非伝統的な方法により保たれている。それは従来の暗号学的な証明を最初から放棄したも同然である。問うべきは可能な攻撃の存在そのものではなく、それが現実的なリスクとなりえるかどうかという点である。この点は、分散分権型の機械学習にも大きな示唆を与える。すなわち、伝統的な暗号学的安全性にも、中央集権的な権威にも頼らず、工学的に妥当な学習手法を設計できるだろうか。次節において、まず、分散分権型における合意問題について見てゆく。

1.4 秘匿集計問題

分散分権型の学習の設定に戻り、式 (1.16) に与えた集計をどのようにデータプライバシーを保存しつつ行えるか考えよう。ここで「データプライバシー」というとき、大きく分けて 2 つの意味がある。ひとつは内部プライバシーとでも呼ぶべきもので、S 人の参加者同士でいかに生データを出さずに学習を行えるか、という問題となる。もうひとつは外部プライバシーとでもいうべきもので、学習したパターン辞書をたとえば第 3 者に外販するときにどの程度元データなり参加者固有の情報を漏らさないか、という意味である。本節で考えるのは前者である。以下では最初にプライバシー制約を考えない簡単化した設定で動的合意法という分権的な集計算法を導入し、次いで、データプライバシーについて制約を満たすための無作為分割法という技術を説明する。動的合意法においては通信グラフのスペクトル構造が本質的役割を果たすので、一例としてサイクルグラフについての解析的な結果を与える。

1.4.1 動的合意法

プライバシーの問題はひとまずわきに置いて、どのように中央のサーバーなしに式 (1.16) の集計問題を解けるか考えよう。

図 1.1 のように、参加者には $s = 1, \ldots, S$ という番号が付けられているとする。ここで通信路を表すために $\mathsf{A} \in \{0,1\}^{S\times S}$ という**隣接行列** (adjacency matrix) が与えられているとする。通信は双方向に行われると想定し、したがって A は対称行列である。図 1.3 は最も単純な通信路であるサイクルグラフの例である。式 (1.16) において N_k はスカラー、$\boldsymbol{T}_k$ はベクトルであるが計算はベクトルの要素ごとに独立に行えるので、以下、スカラー $\{\xi^s\}$ の総和を計算する問題を考えることにする。すなわち

$$\bar{\xi} = \sum_{s=1}^{S} \xi^s = \mathbf{1}_S^\top \boldsymbol{\xi}, \tag{1.26}$$

である。言うまでもなく ξ^s は N_k^s か、または $\boldsymbol{T}_k^s$ のひとつの成分を表し、$\boldsymbol{\xi} \triangleq (\xi^1, \ldots, \xi^S)^\top$ と定義した。また、$\mathbf{1}_S \in \mathbb{R}^S$ は要素がすべて 1 の S 次元ベクトルを表す。

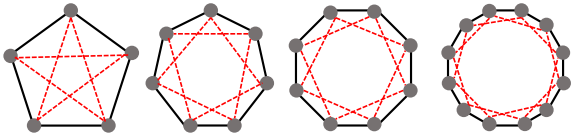

図 1.3 1 階と 2 階のサイクルグラフの例。左から $S = 5, 7, 8, 12$。2 階のサイクルグラフは 1 階のサイクルグラフに点線で表される辺を追加したもの。グラフの定義は本文参照。

各参加者は隣接行列 A でつながった相手と通信できる。通信開始前（時刻 $t = 0$ とする）に第 s 番目の参加者が持っているのは ξ^s というひとつのスカラーだけである。$t = 1$ において参加者らは A でつながった相手に自分の手持ちの値を渡すことができる。各 t において s が持っている値を $\xi^s(t)$ と表すことにする。$\xi^s(0) = \xi^s$ から始めて、次のような更新規則を考えてみる。

$$\xi^s(t+1) = \xi^s(t) + \epsilon \sum_{j=1}^{S} \mathsf{A}_{s,j}[\xi^j(t) - \xi^s(t)], \tag{1.27}$$

ここで ϵ はある（小さい）定数である。S 人すべての参加者がこのような更新を同期的に行うとしよう。行列形式ではこの式 (1.27) は次のように書ける。

$$\boldsymbol{\xi}(t+1) = \mathsf{W}_\epsilon \boldsymbol{\xi}(t) \quad \text{ただし} \quad \mathsf{W}_\epsilon \triangleq \mathsf{I}_S - \epsilon(\mathsf{D} - \mathsf{A}) \tag{1.28}$$

ここで I_S は S 次元の単位行列、D は次数行列と呼ばれる対角行列で、$D_{i,j} \triangleq \delta_{i,j} \sum_{l=1}^{S} A_{i,l}$ のように定義される。$\delta_{i,j}$ はクロネッカーのデルタである。また、$\boldsymbol{\xi}(t) \equiv (\xi^1(t), \ldots, \xi^S(t))^\top$ とおいた。

行列形式で見ると明らかなように、系全体で見ると、1 ラウンド進むごとに W_ϵ という行列を新たに掛けてゆくことになる。何度も何度も行列を掛けてゆくと、値が発散するか、またはどんどん小さくなってしまうような気がするが、そうならない場合がひとつだけある。それは、賭け続ける行列の最大固有値に属する固有値がちょうど 1 になっている場合である。なぜかというとこういうことである。仮定から W_ϵ は実対称行列である。実対称行列の固有値は実数である。固有値を大きい順に $\nu_1, \ldots, \nu_S$ とし、正規直交化された固有ベクトルを $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_S$ としよう。すると、いわゆるスペクトル分解により

$$
\begin{aligned}
\mathsf{W}_\epsilon &= \sum_{i=1}^{S} \nu_i \boldsymbol{u}_i \boldsymbol{u}_i^\top \\
\mathsf{W}_\epsilon^2 &= \sum_{i=1}^{S} \nu_i \boldsymbol{u}_i \boldsymbol{u}_i^\top \sum_{j=1}^{S} \nu_j \boldsymbol{u}_j \boldsymbol{u}_j^\top = \sum_{i=1}^{S}\sum_{j=1}^{S} \nu_i \nu_j (\boldsymbol{u}_i^\top \boldsymbol{u}_j) \boldsymbol{u}_i \boldsymbol{u}_j^\top \\
&= \sum_{i=1}^{S} \nu_i^2 \boldsymbol{u}_i \boldsymbol{u}_i^\top \\
&\cdots \\
\mathsf{W}_\epsilon^\infty &= \sum_{i=1}^{S} \nu_i^\infty \boldsymbol{u}_i \boldsymbol{u}_i^\top
\end{aligned}
$$

のように、W_ϵ の任意回のべきが、固有値のべきに化ける[2]。したがって、最大固有値が $\nu_1 = 1$ で、その他の固有値の絶対値が 1 未満であれば、無限回かけた暁には

$$\boldsymbol{\xi}(\infty) = \mathsf{W}_\epsilon^\infty \boldsymbol{\xi}(0) = \boldsymbol{u}_1 \boldsymbol{u}_1^\top \boldsymbol{\xi}(0)$$

になってしまう。たとえば $\nu_2 = 0.9$ だったとすれば、$0.9^{10} = 0.35$、$0.9^{100} = 2.7 \times 10^{-5}$ のように、掛け続けるとどんどん小さくなり、しまいには消え去ってしまうからである。

では、最大固有値がぴったり 1 などという都合の良いことが本当に成り立っているのだろうか。答えはイエスである。グラフ理論に詳しい人なら、W_ϵ の定義式 (1.28) に出てくる $\mathsf{D} - \mathsf{A}$ が**グラフラプラシアン** graph Laplacian) と呼ばれ、その最小固有値が 0 であること、そして $\mathbf{1}_S$ が固有ベクトルになっていることを知っているであろう。実際、次数行列の定義から

$$[\mathsf{W}_\epsilon \mathbf{1}_S]_i = \sum_{j=1}^{S} (W_\epsilon)_{i,j} 1 = 1 - \epsilon \left(D_{i,i} - \sum_{j=1}^{S} A_{i,j} \right) = 1 \tag{1.29}$$

が成り立つ。したがって $\nu_1 = 1$ で、規格化された固有ベクトルは $\boldsymbol{u}_1 = \frac{1}{\sqrt{S}} \mathbf{1}_S$ である。そしてこのことから、もしも他のあらゆる固有値の絶対値が 1 より小

[2]固有値、スペクトル分解などの数学用語については、ストラング[20] など標準的な線形代数の教科書を参照のこと。

さければ、無限回の通信ののちに、式 (1.28) は

$$\boldsymbol{\xi}^* \triangleq \boldsymbol{\xi}(\infty) = \frac{1}{S}\mathbf{1}_S\,\mathbf{1}_S^\top\boldsymbol{\xi}(0) = \frac{1}{S}\mathbf{1}_S \times \bar{\xi} \tag{1.30}$$

に収束する。言い換えると、どの参加者が持っている値もまったく同じ $\frac{1}{S}\bar{\xi}$、すなわち、求めたい総和の $\frac{1}{S}$ 倍になっている。ゆえ、収束後に S をかければ総和が求められる。力学系の時間発展を模擬するようにして全員の合意を形成したともみられるので、この算法を**動的合意法** (dynamic consensus) と呼んでよい。 局所的なデータのやり取りだけで全員にわたる総和ないし平均を求め、なおかつそれを各人に周知できた（合意を形成できた）ということである。これは協調制御 (cooperative control) の理論や、マルチエージェントシステムの利害調整 (multi-agent coordination) の問題において重要な技術要素になっている。制御理論における動的合意法の由来については、たとえば Ren らのサーベイ論文[15] を参照されたい

話が出来過ぎのように思う読者がいるかもしれないので、ここで実際の数値例を挙げておく。隣接行列として図 1.3 にあるような、階数 2、ノード数 $S = 12$ のサイクルグラフを生成し、$(-10, 10)$ の一様乱数で初期値 $\xi^1(0), \ldots, \xi^{12}(0)$ を与えた。図 1.4 は 12 個の値が、その平均値に収束してゆく様子を示している。大体 20 ラウンドでほとんど完全に収束していることが分かる。

動的合意法は分散分権型の機械学習に非常に適した方法であるが、大きな問題がまだ 2 つ残っている。ひとつはデータプライバシーをどう保証するかという問題であり、もうひとつは通信路グラフの隣接行列をどう設計すべきかという問題である。それぞれ以下に見てゆく。

1.4.2 無作為分解法による秘匿計算

オリジナルの動的合意法の明らかな問題は、初回に自分の持っているデータ ξ^s を近傍の参加者に渡さなければならないことである。2 回目以降は他人のデータと混じるため、渡された値から元の値を復元することは簡単ではなくなるが、初回が問題である。

この問題に対する非常に簡単な解決策がある。それは、一回の反復ラウンドを、N_c 回に分けることである。対応して手元のデータを

$$\xi(t)^s = \xi(t)^{s[1]} + \cdots + \xi(t)^{s[N_c]} \tag{1.31}$$

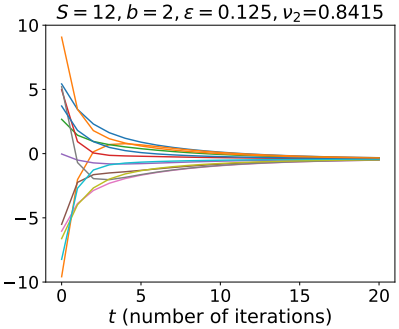

図 1.4　動的合意法による収束の様子。グラフは階数 $b = 2, S = 12$ のサイクルグラフで、$\epsilon = \frac{1}{4b}$ としてある。

のように N_{C} 個の塊に分解する。分解の仕方は各参加者の任意に任せる。動的合意法はデータに対して線形な演算のみから成り立っているので、全ての参加者が同数の塊から成る分解をしている限り

$$\bar{\xi} = \sum_{s=1}^{S} \xi^{s} = \sum_{s=1}^{S} \sum_{h=1}^{N_c} \xi^{s[h]} = \sum_{h=1}^{N_{\mathrm{C}}} \left(\sum_{s=1}^{S} \xi^{s[h]} \right) = \sum_{h=1}^{N_c} \bar{\xi}^{[h]} \tag{1.32}$$

のように、N_{C} 個の塊ごとに総和を計算し、最後にそれを手元で合算すればよい。ただし $\bar{\xi}^{[h]}$ は h 番目の塊に関する総和である。これを**無作為分割法** (random chunking method) と呼ぶ。

ただし、N_c 個の塊に分けたとしても、それらすべてが同じ参加者に送られてしまう場合は意味がない。そのため、理想的には、N_{C} 回のサブ・ラウンドごとに、まったく別の参加者と通信するように隣接行列 A を調整しなければならない。ただし、次数によってはそれは困難ないし実行不可能なので、次善の策として、隣接行列 A をランダムに選択するか、あるいはひとつの A においてノード番号をランダムに入れ替える。その場合、偶然、N_{C} 回続けてある参加者が別の参加者の近傍に当たるということはあり得る。その確率を見積もってみよう。

今、参加者 s のデータを N_c 回続けて受け取ってしまう参加者を j とし、参加者 j を「s の侵害者」と呼ぶことにする。1 回のサブ・ラウンドにおいて、他の参加者が任意の接続をなしうるときに、参加者 j が s の近傍に来る確率は、参加者 s に対応する通信路グラフの次数（通信をすべき参加者数）を d^s とすると

$$\binom{S-2}{d^s-1}\binom{S-1}{d^s}^{-1} = \frac{d^s}{S-1}, \tag{1.33}$$

のように見積もることができる。左辺で分母に来る二項係数は、$S-1$ 人いる s 以外の参加者群の中から、d^s 人の近傍を選択する場合の数を意味する。分子は、j 以外のノードが近傍になる場合の数に対応する。二項係数を出すまでもなく、「$S-1$ 人の中から、d^s 人を選ぶときに、その d^s の中に j が入る確率」と右辺を直接解釈することも可能である。

このことから、j が s の侵害者となる確率は、$\left(\frac{d^s}{S-1}\right)^{N_\mathrm{C}}$ のように書ける。他の $S-2$ 人の一部が侵害者になりえることを考えると、ブールの不等式から、s のデータプライバシーが侵害 (breach) される確率について

$$p^s_{\mathrm{breach}} \le \sum_{j\neq s}\left(\frac{d^s}{S-1}\right)^{N_\mathrm{C}} = (S-1)\left(\frac{d^s}{S-1}\right)^{N_\mathrm{C}} \tag{1.34}$$

のような評価ができる。したがって任意の参加者のデータプライバシーが保たれる確率は

$$p_{\mathrm{secure}} > \prod_{s=1}^{S}\left\{1-(S-1)\left(\frac{d^s}{S-1}\right)^{N_\mathrm{C}}\right\} \ge 1 - S(S-1)\left(\frac{d_{\max}}{S-1}\right)^{N_\mathrm{C}} \tag{1.35}$$

を満たす。ただし $d_{\max}$ はすべての参加者の中で最大の次数であり、第 2 の不等式はベルヌーイの不等式による。通信路は一般的に疎なグラフとなるように選ばれるので、$\frac{d_{\max}}{S-1} \ll 1$ が成り立つ。ゆえ、N_C がそこそこ大きければ、最右辺は 1 に近い。つまり、100% 安全ではないとしても、ほとんど常に安全と言えるようにできるということである。この結果を定理としてまとめておこう。

定理 1.1（無作為分割法のデータプライバシー保障） 動的合意法と無作為分割法を組み合わせた合意算法において、データプライバシーが侵害される確率

Algorithm 2 無作為分割法を使った動的合意

入力: ϵ、N_c、A
初期化: $\bar{\xi} = 0$.
それぞれの参加者が自分のデータ ξ^s を N_c 個の塊に分割する。
for $h \leftarrow 1, \ldots, N_c$ **do**
    ルーターがランダムに参加者番号を振り直すことで A を組み直す。
    repeat
        式 (1.27) により各 s での値を更新する。
    until 収束
    $\bar{\xi} \leftarrow \bar{\xi} + \bar{\xi}^{[h]}$
end for

は式 (1.34) のような上限を持つ。適切に N_{C} と通信路を選ぶことにより、侵害確率を指数関数的に 0 に近づけることができる。

この式から分かる通り、塊の数 N_{C} に対して指数関数的に安全性は上がってゆく。なおかつ、疎な、したがって次数の低い通信路を使うと安全性がさらに増すことが分かる。この状況はビットコインの承認者選択における安全性保障とやや似ている。我々は 100% 確実な安全性を保障することはできないが、うまく N_{C} と通信路を選べば、きわめて高い確率で安全な算法を設計することができる。Algorithm 2 に、無作為分割法に基づく動的合意法の計算手順をまとめておく。

1.4.3 サイクルグラフの固有値

動的合意法の収束は、行列 W_ϵ の第 1 番目と第 2 番目の固有値の比にかかっている。疎なグラフの具体例として、図 1.3 に示したような、S 頂点を持ち階数が b の**サイクルグラフ** (cycle graph) を考えてみよう。このグラフを C_S^b という記号で表すことにする。これは**正則グラフ** (regular graph)、すなわちすべての頂点が同一の次数 $d = 2b$ 持つグラフである。このグラフは、階数を低く保つことで疎なグラフとなり、式 (1.35) で示した通り、内部プライバシーが侵害される確率を低く抑えることができる。また、全ての頂点の次数が同じという意味で、民主主義的なグラフとも言える。

隣接行列 A の固有値と固有ベクトル

W_ϵ の固有値は隣接行列 A の固有値から直ちに求められるので、サイクルグラフの隣接行列の固有値を求めてみよう。そのためには、隣接行列のベクトル表現を導入すると便利である。S 次元のユークリッド空間の正規直交基底 $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_S$ を使うと、隣接行列は $\mathsf{A} = \sum_{i,j} \mathsf{A}_{i,j} \boldsymbol{e}_i \boldsymbol{e}_j^\top$ のように書けるから、C_S^b の隣接行列は

$$\mathsf{A}(\mathrm{C}_S^b) = \sum_{s=1}^{S} \sum_{j=1}^{b} (\boldsymbol{e}_{s+j} \boldsymbol{e}_s^\top + \boldsymbol{e}_{s-j} \boldsymbol{e}_s^\top) \tag{1.36}$$

である。ただし b は $S \geq 2b+1$ を満たし、すべての添え字には周期的境界条件を課す。すなわち、$\boldsymbol{e}_i$ の添え字が $1, \ldots, S$ の範囲から出る場合は、S の整数倍を加えるか引くかして、$1, \ldots, S$ の範囲の値と同一視する。たとえば $j = 0$ は S と同じで、$j = S+1$ は 1 と同一視する。一般に、周期 S の 1 次元格子上の任意の関数 $u(s)$ は離散フーリエ変換による表現

$$u(s) = \sum_{m=1}^{S} U_m \phi_m(s) \tag{1.37}$$

を持つ。ここで U_m は m 番目の周波数に対するフーリエ係数である。$\phi_m(s)$ は m 番目の周波数に対応する基底関数であり、

$$\phi_m(s) \triangleq \frac{1}{\sqrt{S}} \mathrm{e}^{\mathrm{i}\omega_l(s-1)}, \quad \omega_m \triangleq \frac{2\pi}{S}(m-1) \tag{1.38}$$

のように定義される。i は虚数単位である。この基底が、$\phi_m(s+S) = \phi_m(s)$ を満たすことは容易に確認できる。

上に定義した 1 次元周期格子上の関数 $u(s)$ は、$\boldsymbol{u} = \sum_{s=1}^{S} u_s \boldsymbol{e}_s$ によって、S 次元ユークリッド空間のベクトルと 1 対 1 に対応する。これに $\mathsf{A}(\mathrm{C}_S^b)$ を作用させ、どのように $\{U_m\}$ を決めると A の固有ベクトルになるか考えてみよう。

$$\begin{aligned} \mathsf{A}(\mathrm{C}_S^b)\boldsymbol{u} &= \sum_{s=1}^{S} \sum_{j=1}^{b} \sum_{m=1}^{S} U_m \phi_m(s) (\boldsymbol{e}_{s-j} + \boldsymbol{e}_{s+j}) \\ &= \sum_{s=1}^{S} \sum_{j=1}^{b} \sum_{m=1}^{S} U_m \phi_m(s) (\mathrm{e}^{\mathrm{i}\omega_m j} + \mathrm{e}^{-\mathrm{i}\omega_m j}) \boldsymbol{e}_s \end{aligned} \tag{1.39}$$

$$= \sum_{s=1}^{S} \boldsymbol{e}_s \sum_{m=1}^{S} U_m \phi_m(s) \sum_{j=1}^{b} 2\cos(\omega_m j) \tag{1.40}$$

が成り立つことが分かる。$\boldsymbol{u}$ が固有ベクトルになるためには、右辺が $\boldsymbol{u}$ に比例しなければならない。このための条件は

$$\left[\lambda - \sum_{j=1}^{b} 2\cos(\omega_m j) \right] U_m = 0 \tag{1.41}$$

である。これと規格化条件[3]を満たすように $\lambda, \{U_m\}$ を決めるのは簡単である。例えば $U_1 = 1$ とし他を全部 0 にすれば、$\lambda = \sum_{j=1}^{b} 2\cos(\omega_1 j)$ においてすべての m に対し上式は満たせる。同じく、$U_2 = 1$ とし他を全部 0 にすれば、$\lambda = \sum_{j=1}^{b} 2\cos(\omega_2 j)$ においてすべての m に対し上式は満たせる。$U_l = 1$ で他を 0 とした場合に対応する固有ベクトルと固有値をそれぞれ $\boldsymbol{u}_l, \lambda_l$ とすると、$l = 1, \ldots, S$ に対し

$$\boldsymbol{u}_l = \sum_{s=1}^{S} \boldsymbol{e}_s \phi_l(s) = \sum_{s=1}^{S} \boldsymbol{e}_s \frac{1}{\sqrt{S}} \exp\left\{ \frac{2\pi \mathsf{i}}{S}(l-1)(s-1) \right\} \tag{1.42}$$

$$\lambda_l = \sum_{j=1}^{b} 2\cos(\omega_l j) = \sum_{j=1}^{b} 2\cos\left\{ \frac{2\pi}{S}(l-1)j \right\} \tag{1.43}$$

となることが分かる（一般に固有値の降順に整列されていないことに注意）。これが $\mathsf{A}(C_S^b)$ の固有ベクトルと固有値である。最大の固有値は明らかに $l = 1$ において生じ、$\boldsymbol{u}_1 = \frac{1}{\sqrt{S}}\mathbf{1}_S, \lambda_1 = 2b$ であることが分かる。

W_ϵ の固有値と動的合意法の収束

さて隣接行列の固有ベクトルが分かったので W_ϵ の固有ベクトル・固有値について考えよう。サイクルグラフ C_S^b はすべての頂点が $d = 2b$ という次数を持つ正則グラフなので、$\mathsf{W}_\epsilon = \mathsf{I}_S - \epsilon(\mathsf{D} - \mathsf{A}) = (1 - \epsilon d)\mathsf{I}_S + \epsilon\mathsf{A}$ が成り立つ。任意のベクトルは単位行列の固有ベクトルになっているので、式 (1.42) の $\{\boldsymbol{u}_l\}$ は W_ϵ の固有ベクトルにもなっていることが分かる。式 (1.42) に対応し

[3]取りえる範囲が複素数に拡張されているので、規格化条件は $\boldsymbol{u}$ の共役転置 $\boldsymbol{u}^\dagger$ により、$1 = \boldsymbol{u}^\dagger \boldsymbol{u} = \sum_{s=1}^{S} u^*(s)u(s) = \sum_{s=1}^{S} |u(s)|^2$ のように書かれる。ただし $u(s)^*$ は $u(s)$ の複素共役、$|u(s)|$ は（複素数の）絶対値を表す。

て、W_ϵ の（必ずしも降順に整列されていない）固有値を $\tilde{\nu}_1, \ldots, \tilde{\nu}_S$ と置く。式 (1.42) から単位行列の項の分の変更を受け

$$\tilde{\nu}_l = 1 - \epsilon d + \epsilon \lambda_l \tag{1.44}$$

$$= 1 - 4\epsilon \sum_{j=1}^{b} \sin^2 \left\{ \frac{\pi}{S}(l-1)j \right\} \tag{1.45}$$

となっていることが分かる。最大の固有値 ν_1 は再右辺の $\sin^2$ の項が最小値 0 を取るところで生ずる。これは明らかに $\nu_1 = \tilde{\nu}_1 = 1$ である。第 2 固有値は $l = 1$ および $l = S - 1$ に対応して

$$\nu_2 = 1 - 2\epsilon \sum_{j=1}^{b} \left(1 - \cos \frac{2\pi j}{S} \right) = 1 - 4\epsilon \sum_{j=1}^{b} \sin^2 \frac{\pi j}{S} \tag{1.46}$$

となっていることが分かる。これが絶対値においても第 2 番目に大きいかどうかは ϵ の値に依存する。式 (1.45) より、ν_l は $1 - 4\epsilon b$ を下回ることができないから、たとえば $\epsilon = \frac{1}{4b} = \frac{1}{2d}$ と選べばすべての固有値は非負になり、したがって ν_2 が絶対値の意味でも第 2 番目に大きい。

1.4.4 サイクルグラフにおける動的合意法の収束

このようにサイクルグラフでは W_ϵ の固有値が解析的に求められるので、動的合意法の収束のだいたいの傾向をつかむために有用である。実用上のひとつの興味は、参加者の人数 S が大きいとき収束が非常に遅くなったりしないか、というものである。動的合意法の第 t 回目の反復において、$\left(\frac{\nu_2}{\nu_1}\right)^t$ が小さければ小さいほど収束は早い[4]。今の場合、総和 $\bar{\xi}$ を求めるのが目的である。式 (1.30) によれば、反復回数 t が $t \to \infty$ となると

$$\|\sqrt{S}\mathbf{W}_\epsilon{}^t \boldsymbol{\xi}(0)\|_2 \to \left\| \frac{1}{\sqrt{S}} \mathbf{1}_S \bar{\xi} \right\|_2 = |\bar{\xi}|$$

が成り立つ。$\|\cdot\|_2$ は ℓ_2 ノルムである。それゆえ、左辺と再右辺の差を $\bar{\xi}$ で割って作られる

[4] $\nu_1 = 1$ なので冗長な表現だが、ν_2 そのものではなくて ν_2 との違いが重要なのでこのように書いた。

$$e_\epsilon(t) \triangleq \frac{1}{|\bar{\xi}|}\sqrt{\|\sqrt{S}\mathbf{W}_\epsilon{}^t\boldsymbol{\xi}(0)\|_2^2 - \bar{\xi}^2} \tag{1.47}$$

は我々の文脈における相対誤差として自然な選択である。平方根の中は正であることに注意。ここで W_ϵ の降順に並べた固有値を $\nu_1, \ldots, \nu_S$ とし、ϵ を適切に調整することで ν_2 が絶対値の意味でも第 2 番目に大きいと仮定しよう。 ν_2 に対応する規格化された固有ベクトルを $\boldsymbol{u}_2$ とすれば、$t \to \infty$ とともに漸近的に

$$e_\epsilon(t) \to \sqrt{S}\left(\frac{\nu_2}{\nu_1}\right)^t \times \frac{\boldsymbol{u}_2^\top\boldsymbol{\xi}(0)}{\bar{\xi}} \sim \sqrt{S}\left(\frac{\nu_2}{\nu_1}\right)^t \times \mathrm{O}(1) \tag{1.48}$$

のようになる。それゆえ、$\sqrt{S}(\nu_2/\nu_1)^t$ を相対誤差の代用品として使うことができる。参加者数 S が大きいときに、これがある小さな値 δ より小さくなるために必要な反復回数 t を求めよう。式 (1.46) の総和は、$S \gg \pi b$ であるとき、正弦関数をテイラー展開することで足し上げることができる。すなわち

$$\left(\frac{\nu_2}{\nu_1}\right)^t \approx 1 - \frac{2\pi^2}{3S^2}\epsilon t b(b+1)(2b+1). \tag{1.49}$$

である。この解析的な近似表現を使うと、相対誤差が δ を下回るために必要な最低の回数を

$$t \approx \frac{3S^2\ln(\sqrt{S}/\delta)}{2\pi^2\epsilon b(b+1)(2b+1)} = O\left(\frac{S^2\ln(\sqrt{S}/\delta)}{\epsilon b^3}\right). \tag{1.50}$$

と見積もることができる。つまり参加者数 S の 2 乗の程度で収束に必要な反復回数は増える。サイクルグラフについての以上の結果をひとまずまとめておこう。

定理 1.2（サイクルグラフでの動的合意法の収束） 階数 b、頂点数 S のサイクルグラフで、$\nu_2 > |\nu_S|$ となるよう ϵ が選ばれていれば、動的合意法 (1.27) は総和 $\bar{\xi}$ に収束する。相対誤差が δ を下回るために必要な反復回数 t は $t \sim O\left(\frac{S^2\ln(\sqrt{S}/\delta)}{\epsilon b^3}\right)$ である。

算法設計の常識からすれば、グラフの頂点数の 2 乗に比例する反復回数は遅い部類に入るであろう。だとすれば、サイクルグラフ以外のグラフを選ぶこと

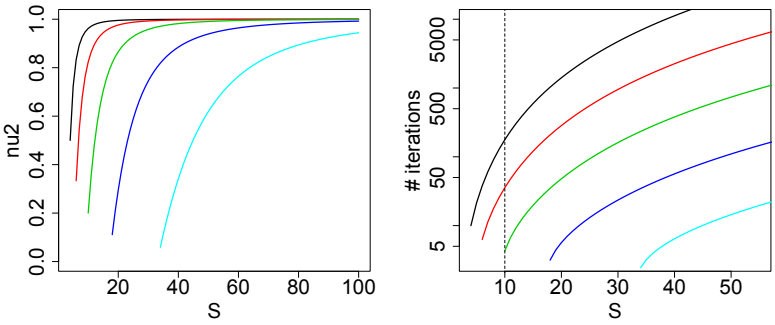

図 1.5 左: サイクルグラフ C_S^d の第 2 固有値 ν_2 を S の関数としてプロットしたもの。右: $(\nu_2/\nu_1)^t = 10^{-3}$ に到達するまでの反復数の様子。どちらの図も、左から右に、$b = 1, 2, 4, 8, 16$ に対応している。

でこれを改善できないか、という発想が生まれてくる。これは**スペクトルグラフ理論** (spectral graph theory) の興味深い問題に関係しており、かなりのことが分かっている。次節で軽くさわりを紹介しよう。

本節の残りでは、サイクルグラフの収束にまつわる数値例をいくつか挙げておこう。図 1.5 の左側の図は、式 (1.46) を元に、ν_2 を C_S^d の頂点数 S の関数として示したものである。左から、$b = 1, 2, 4, 8, 16$ に対応する。図から分かる通り、ν_2 は S の大きい側で急速に 1 に近づいてしまい、動的合意法の収束に困難を生ずることが分かる。予期される通り、b が大きいとこの傾向は弱まり、したがって収束は速まる。一方、図 1.5 の右側は $(\nu_2/\nu_1)^t = 10^{-3}$ に至るまでの反復回数 t を表している。例えば $S = 10$ のとき、階数最低のサイクルグラフ $b = 1$ は 236 回の反復が必要で、$b = 2$ と 3 ではそれぞれ 47 回と 6 回の反復が必要であることが分かる。なお、$S = 12, b = 2$ に対する図 1.4 では、$\nu_2 = 0.8415$ ゆえ $(\nu_2/\nu_1)^{20} = 0.031$、$\sqrt{S}(\nu_2/\nu_1)^{20} = 0.110$ である。

どの程度の誤差を許容できるかは、動的合意法が使われる学習の算法にも依存する。もし学習法がデータのばらつきに敏感なようなものであれば、EM 法

の内部で動的合意法を使う際には収束の様子に細心の注意を払う必要がある。単発の動的合意における収束ではなく、分散分権型の学習と組み合わせた場合の収束の管理の方法は必ずしもよく分かっているわけではないので、問題ごとに個別に考えてゆく必要がある。

1.4.5　修正サイクルグラフとその意義

さて、通信路の設計による動的合意法の収束改善という問題に戻ろう。サイクルグラフから出発した時の興味深い通信路の例として、逆弦つきサイクルグラフ (cycle with inverse chords) という特殊なグラフが知られている[18]。これは C_S^1 の各頂点 s から、$(s-1)(j-1)=1 \mod S$ を満たす相手に追加の辺を張る、というだけのものである。これは次数 $d=3$ の正則グラフとみなせる[5]。隣接行列 A の i 番目に大きい固有値を λ_i としよう。このグラフにおいては、興味深いことに、$\Delta \triangleq \lambda_1 - \lambda_2$ という量に下限が存在することが知られている。その下限は S にはほとんど依存せず、主に次数で決まる。次数 d の正則グラフに対して成り立つ式 (1.44) より

$$\nu_1 - \nu_2 = \epsilon(\lambda_1 - \lambda_2) \geq \epsilon\Delta_{\min} \tag{1.51}$$

が言える。したがってこのグラフでは

$$\frac{\nu_2}{\nu_1} \leq 1 - \epsilon\Delta_{\min} \tag{1.52}$$

となっていることが分かる。ただし Δ の下限を $\Delta_{\min}$ と表した。先に見た通り、ν_2/ν_1 は、（正の範囲で）小さければ小さいほど収束が速い。したがって、このグラフでは、適切な ϵ を選ぶことで、非常に早い収束を達成できる可能性がある。

前節と同様に考えると、このグラフでは、相対誤差 δ を下回るために必要な反復回数を

---

[5] 本来の定義では S は素数に限られるが、それは気にせず任意の S にこの規則を採用する。この定義に基づくと、追加の弦が張れない場合がありえるので、一般にこのグラフは正則グラフとはならない。しかし「ほとんど」正則グラフということは言えるので、正則グラフに対して成り立つ不等式 (1.51) および (1.52) を使って収束の評価を行っている。

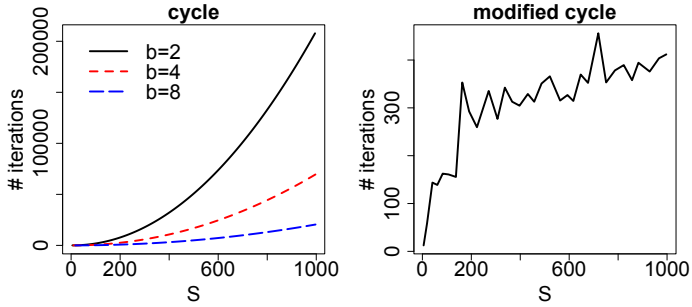

図 1.6 $\sqrt{S}\left(\frac{\nu_2}{\nu_1}\right)^t = 10^{-3}$ を達成するために必要な反復数。

$$t \sim O\left(\frac{\ln(\sqrt{S}/\delta)}{|\ln(1-\epsilon\Delta_{\min})|}\right) \tag{1.53}$$

のように見積ることができる。上記の結果を定理 1.2 と比較してみるのは興味深い。上に述べた通り、$\Delta_{\min}$ は S にほとんど依存しないので、反復回数 t は、頂点数（参加者数）S に対し、おおむね $\ln S$ で依存することになる。これは人数が多くても収束速度の悪化はほとんどない、ということを意味する。

これはやや直感に反する気がしなくもないので、ここで実際に数値的に収束速度を確認してみることにしよう。$\sqrt{S}(\nu_2/\nu_1)^t = 10^{-3}$ を達成するために必要な反復回数 t を計算してみる。結果を図 1.6 に示す。公平な比較のため、グラフの最大次数を $d_{\max}$ としたときに、$\epsilon = 1/d_{\max}$ のように ϵ を選んだ。左が通常のサイクルグラフ、右が修正サイクルグラフの結果である。サイクルグラフの場合、定理 1.2 で見た通り、収束に必要な反復数はおおむね放物線的に増えていることが分かる。一方、修正サイクルグラフの場合、まず縦軸の目盛が段違いで小さく、かつ、その上昇の度合いも非常に遅い（つまり会員数が増えても収束に必要な反復数があまり増えていない）ことが分かる。

もうひとつ修正サイクルグラフで興味深いのは、反復数に不連続な増減が見られることである。一部これは、脚注 5 で述べたような定義の拡張に由来するが、常にそうであるわけではない。この種類のグラフの研究はスペクトルグラ

フ理論の中心課題のひとつであり、研究の余地がある。この点については最近の論文[9] も参照されたい。

1.5　スパース混合ガウスモデルによる分権分散型学習

1.2.1 節で述べたモデルにおいて、観測モデルの式 (1.2) における $f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k)$ と、事前分布の式 (1.5) における $p(\boldsymbol{\theta}_k)$ については具体的なモデルを指定していなかった。この節では指数分布族の代表例として、特に多次元ガウスモデルを選んだ時の具体的な算法について考える。このモデルの最尤推定は graphical LASSO という算法を利用することで効率よく行える。この算法の導出も詳しく説明する。

1.5.1　モデルの設定

観測モデルとしての多次元ガウスモデルの分布関数を明示的に書くと

$$\begin{aligned} f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k) &= \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, (\Lambda_k)^{-1}) \\ &= \frac{\det(\Lambda_k)^{\frac{1}{2}}}{(2\pi)^{\frac{M}{2}}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}^s - \boldsymbol{\mu}_k)^\top \Lambda_k (\boldsymbol{x}^s - \boldsymbol{\mu}_k)\right\} \end{aligned} \tag{1.54}$$

である。モデルのパラメターとしては $\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k, \Lambda_k\}$ ということである。$\det(\cdot)$ は、カッコ内の行列の行列式を計算することを意味する。$\boldsymbol{\mu}_k \in \mathbb{R}^M$ は平均、 $\Lambda_k \in \mathbb{R}^{M\times M}$ は**精度行列** (precision matrix) を表す。これらのパラメターへの事前分布として、ひとつの実用的な選択は次のガウス-ラプラス分布である。

$$\begin{aligned} p(\boldsymbol{\theta}_k) = p(\boldsymbol{\mu}_k, \Lambda_k) &\propto \mathcal{N}(\boldsymbol{\mu}_k \mid \mathbf{0}, (\lambda_0 \mathsf{I}_M)^{-1}) \exp\left(-\frac{\rho_0}{2}\|\Lambda_k\|_1\right) \\ &= \left(\frac{\lambda_0}{2\pi}\right)^{\frac{M}{2}} \exp\left(-\frac{\lambda_0}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\mu}_k - \frac{\rho_0}{2}\|\Lambda_k\|_1\right) \end{aligned} \tag{1.55}$$

新しい定数 ρ_0, λ_0 が出てきたが、これらは事前に与えられているとみなす。$\|\cdot\|_1$ は一般に ℓ_1 ノルムを表しており、今の場合は

$$\|\Lambda_k\|_1 = \sum_{i=1}^{M}\sum_{j=1}^{M} |(\Lambda_k)_{i,j}| \tag{1.56}$$

のように、行列要素の絶対値を全部足したものである。先ほども述べたが、事前分布は不必要な偏見をモデルに導入するためのものではなく、解の数値的な性質を改善するために使われている。$\boldsymbol{\mu}_k$ については $\mathbf{0}$ の周りにゆるく拘束することで、外れ値に過度に敏感に反応しないようにしている。同様に Λ_k についても、その行列要素がそれぞれ独立にゼロの周りに分布しているという想定を入れている。これは、強い根拠がない限り、行列要素の値はゼロとみなす、というような約束にしたことにあたる。確率分布の言葉で書いているものの、やっていることはいわゆる正則化項 (regularization term) を入れているのと同じである。この点は後で明らかになる。

1.5.2　対数尤度の表式とパラメター推定

指数型分布族について、パラメター推定を行うための一般式を式 (1.17) に与えた。その式を使う際の実用上の問題は、$G, H, \boldsymbol{\eta}, \boldsymbol{T}$ による表現が必ずしも一般的ではないことである。多次元ガウス分布の場合は、通常使われるパラメターは平均と共分散行列という 2 つだけで簡明であるが、$G, H, \boldsymbol{\eta}, \boldsymbol{T}$ はこれらとの間で相当複雑な関係を持つ。それゆえここでは、理論的な一般式 (1.17) から離れ、改めて最尤推定のためのパラメター推定式を書き下してみる。

式 (1.9) で与えた対数尤度の下限を、$\{\boldsymbol{\mu}_k, \Lambda_k\}$ に関係する部分について書き出してみよう。

$$L(\boldsymbol{\Pi}, \boldsymbol{\Theta}) = \text{c.} + \sum_{k=1}^{K} \ln p(\boldsymbol{\mu}_k, \Lambda_k) + \sum_{s=1}^{S} \sum_{n=1}^{N^s} \sum_{k=1}^{K} r_k^{s(n)} \ln \left[\pi_k^s f(\boldsymbol{x}^s \mid \boldsymbol{\theta}_k)\right]$$

ただし c. はパラメター $\{\boldsymbol{\mu}_k, \Lambda_k\}$ に依存しない定数である。最後の項では $\sum_{\boldsymbol{z}^{s(n)}} q(\boldsymbol{z}^{s(n)}) z_k^s = r_k^{s(n)}$ を使った。 さらに具体的なモデル (1.54) および (1.55) を代入すると

$$L(\boldsymbol{\Pi}, \boldsymbol{\Theta}) = \text{c.} + \sum_{k=1}^{K} \left[-\frac{\lambda_0}{2} \boldsymbol{\mu}_k^\top \boldsymbol{\mu}_k - \frac{\rho_0}{2} \|\Lambda_k\|_1 \right.$$
$$\left. + \frac{1}{2} \sum_{s,n} r_k^{s(n)} \left\{ \ln \det(\Lambda_k) - (\boldsymbol{x}^{s(n)} - \boldsymbol{\mu}_k)^\top \Lambda_k (\boldsymbol{x}^{s(n)} - \boldsymbol{\mu}_k) \right\} \right]$$

となる。ここで式 (1.13) で定義した $N_k^s \triangleq \sum_{n=1}^{N^s} r_k^{s(n)}$ に加え

$$\boldsymbol{m}_k^s \triangleq \sum_{n=1}^{N^s} r_k^{s(n)} \boldsymbol{x}^{s(n)} \quad \mathsf{C}_k^s \triangleq \sum_{n=1}^{N^s} r_k^{s(n)} \boldsymbol{x}^{s(n)} \boldsymbol{x}^{s(n)\top} \tag{1.57}$$

という量を定義する。これらは各参加者の手元で計算できる総和計算である。さらにこれらを参加者にわたり合算する

$$\bar{N}_k \triangleq \sum_{s=1}^{S} N_k^s, \quad \bar{\boldsymbol{m}}_k \triangleq \sum_{s=1}^{S} \boldsymbol{m}_k^s, \quad \bar{\mathsf{C}}_k \triangleq \sum_{s=1}^{S} \mathsf{C}_k^s \tag{1.58}$$

という量も定義しておく。これらの量を使って $L(\boldsymbol{\Pi}, \boldsymbol{\Theta})$ を整理する。任意の列ベクトル $\boldsymbol{a}$ と行列 A について、積が適切に定義できる限り $\boldsymbol{a}^\top \mathsf{A} \boldsymbol{a} = \sum_{i,j} a_i A_{i,j} a_j = \mathrm{Tr}(\mathsf{A} \boldsymbol{a} \boldsymbol{a}^\top)$ が常に成り立つことを利用して

$$\begin{aligned} L(\boldsymbol{\Pi}, \boldsymbol{\Theta}) = \mathrm{c.} + \frac{1}{2} \sum_{k=1}^{K} \big[& -\lambda_0 \boldsymbol{\mu}_k^\top \boldsymbol{\mu}_k - \rho_0 \|\Lambda_k\|_1 \\ & + \bar{N}_k \ln\det(\Lambda_k) - \mathrm{Tr}\left(\Lambda_k (\bar{\mathsf{C}}_k + \bar{N}_k \boldsymbol{\mu}_k \boldsymbol{\mu}_k^\top)\right) + 2\,\mathrm{Tr}\left(\Lambda_k \bar{\boldsymbol{m}}_k \boldsymbol{\mu}_k^\top\right) \big] \end{aligned}$$

となることが分かる。

十分見やすい形になったので、ここで $L(\boldsymbol{\Pi}, \boldsymbol{\Theta})$ を最大化する $\boldsymbol{\mu}_k$ と Λ_k を求めてみよう。$\boldsymbol{\mu}_k$ について微分して $\mathbf{0}$ と等置することで容易に

$$\boldsymbol{\mu}_k = \frac{1}{\lambda_0 + \bar{N}_k} \bar{\boldsymbol{m}}_k \tag{1.59}$$

を得る[6]。精度行列 Λ_k については通常の意味では微分不可能な $\|\Lambda_k\|_1$ という項があるため、$\boldsymbol{\mu}_k$ ほど簡単にはいかない。上の $L(\boldsymbol{\Pi}, \boldsymbol{\Theta})$ の中で Λ_k に関する項を単に拾うと、対数尤度（の下限）を最大化するために解くべきなのは

$$\Lambda_k = \arg\max_{\Lambda_k} \left\{ \ln\det(\Lambda_k) - \mathrm{Tr}(\Lambda_k \Sigma_k) - \frac{\rho_0}{N_k} \|\Lambda_k\|_1 \right\} \tag{1.60}$$

であることがわかる。ただし

$$\Sigma_k \triangleq \frac{1}{N_k} \bar{\mathsf{C}}_k + \boldsymbol{\mu}_k \boldsymbol{\mu}_k^\top \tag{1.61}$$

と定義した。この中の $\boldsymbol{\mu}_k$ は式 (1.59) ですでに求まっていると想定される。

[6] 行列とベクトルについての微分公式は井手[22] の巻末付録を参照のこと。

以上の結果を 1.2.3 節における一般論と対比すると次のようになる。

- 局所的総和: $\{N_k^s, \boldsymbol{m}_k^s, \mathsf{C}_k^s\}_{k=1}^K$ の計算。すなわち、指数型分布族の $\boldsymbol{T}$ という量は、$\{\boldsymbol{m}_k^s, \mathsf{C}_k^s\}$ に対応する。
- 合意形成: 式 (1.58) における $\{\bar{N}_k, \bar{\boldsymbol{m}}_k, \bar{\mathsf{C}}_k\}_{k=1}^K$ の（分散分権型の）計算。
- 最適化: 式 (1.59) による $\{\boldsymbol{\mu}_k\}_{k=1}^K$ の計算と、式 (1.60) による $\{\mathsf{\Lambda}_k\}_{k=1}^K$ の計算。

残る問題は、式 (1.60) をどう解くかである。この問題は ℓ_1 正則化回帰問題との類似性から、**グラフィカルラッソ** (graphical LASSO) などと呼ばれている[5]。LASSO は公式には least absolute shrinkage and selection operator ということになっているが、実際にはこれはむしろ、縄をぎりぎりと締め上げるように回帰係数をゼロに近づける様子が、カウボーイの投げ縄 (lasso) に似ているという直感的理解が先にあったようである[26]。「グラフィカル」の方は、式 (1.60) が、**ガウス型グラフィカルモデル** (Gaussian graphical model) の議論の中で最初に現れたという経緯による。ガウス型グラフィカルモデルというのは、変数間の依存関係の構造をデータから学習する**構造学習** (structure learning) と呼ばれる分野の最も基本となるモデルのひとつで、変数間の依存構造が精度行列と 1 対 1 に対応している。より具体的に言えば、x_i と x_j の間の偏相関係数 (partial correlation coefficient) $r_{i,j}$ は

$$r_{i,j} = -\frac{\Lambda_{i,j}}{\sqrt{\Lambda_{i,i}\Lambda_{j,j}}} \tag{1.62}$$

のように精度行列と関係している。したがって精度行列 $\{\mathsf{\Lambda}_k\}$ を求めることは、変数間の依存関係を表すグラフを求めることである。このあたりの初等的な解説は井手[21] を参照されたい。

1.5.3　graphical LASSO による精度行列の推定

式 (1.60) は k により完全に独立に扱えるので、この節では添え字を落とし、かつ $\rho \triangleq \rho_0/N_k$ として

$$\mathsf{\Lambda} = \arg\max_{\mathsf{\Lambda}} \{\ln\det(\mathsf{\Lambda}) - \mathrm{Tr}(\mathsf{\Lambda\Sigma}) - \rho\|\mathsf{\Lambda}\|_1\} \tag{1.63}$$

の解き方を考えよう。$\mathsf{\Sigma}$ は既知の $M \times M$ 実対称行列とする。これを解く際何に困ったかといえば、$\|\mathsf{\Lambda}_k\|_1$ が普通の意味では微分できないことであった。絶

対値関数にはカドがある。カドの位置では微分係数がうまく定義できない。だから「微分してゼロとおく」という極大値を求めるためのお決まりの技がそのままで使えない。しかしカドの 1 点を除けばまったく問題なく微分できてしまうので、すべてをあきらめるのもやりすぎな気もする。そこで一般に、実数値 x に対し、$\frac{\mathrm{d}|x|}{\mathrm{d}x}$ の代用品として

$$\mathrm{sign}(x) \triangleq \begin{cases} 1, & x > 0 \\ (-1, 1) \text{ の間の何かの値}, & x = 0 \\ -1, & x < 0 \end{cases} \tag{1.64}$$

のような関数を定義する。これを符号関数 (sign function) と呼ぶ。「何かの値」とは無責任に聞こえる定義であるが、話は逆である。むしろ、これは方程式が矛盾しないようにする隠し玉のようなものである。実際、カドにおける値を上記のような不定係数に繰り込んで何とかする一連の手法を、**劣勾配法**（subgradient method）と呼ぶ。ざっくり言えば、これは「微分してゼロとおく」という話と同じなのだが、絶対値の微分が出てくるところでは微分演算を符号関数で置き換えるという手続きのことである。

最適性の条件

ということで勇気を出して、目的関数を Λ で微分してゼロと置いてみよう。行列の微分に関するよく知られた公式[22]

$$\frac{\partial}{\partial \Lambda} \ln \det(\Lambda) = \Lambda^{-1}, \quad \frac{\partial}{\partial \Lambda} \mathrm{Tr}(\Lambda \Sigma) = \Sigma \tag{1.65}$$

を使うと、式 (1.63) の目的関数の勾配をゼロとおいて、形式的に

$$\mathbf{0} = \frac{\partial f}{\partial \Lambda} = \Lambda^{-1} - \Sigma - \rho \, \mathrm{sign}(\Lambda) \tag{1.66}$$

という最適性の条件が得られる。ただし $\mathrm{sign}(\Lambda)$ は、(i, j) 要素が $\mathrm{sign}(\Lambda_{i,j})$ で与えられる行列である。

$\rho = 0$ の場合、この方程式を解くのは簡単である。$\Lambda^{-1} = \Sigma$ より、解は $\Lambda = \Sigma^{-1}$ となる。しかしこれはわれわれの欲しいものではない。第 1 に数値計算上の問題がある。Σ はその定義式 (1.57)-(1.58) および (1.61) を見ると分かる通り、標本共分散という意味を持つ。実用上、次元 M が数十を超えると、

数値計算上しばしば $\mathsf{\Sigma}$ の階数 (rank) に欠損が起こり、逆行列を明示的に求められない。第 2 に、疎な $\mathsf{\Lambda}$ が得られないという問題がある。数値計算上の問題だけなら、疑似逆行列[20] を求めたり、不完全コレスキー分解 (incomplete Cholesky decomposition)[4] を使って解決は可能だが、そうして求まる $\mathsf{\Lambda}$ は疎にならない。後ほど示すが、$\rho > 0$ を与えることで（やや意外なことに）$\mathsf{\Lambda}$ の要素には多くの 0 が現れる。$\mathsf{\Lambda}$ の各要素は変数同士の直接相関を表しているのだから、ノイズかもしれない要素は無視して真に強い信号のみに注目する、という「割り切り」ができることを意味する。これが実用上うれしい点である。

行列のブロック分割

$\rho > 0$ の場合、例の符号関数のおかげで式 (1.65) を解くのは簡単ではなくなる。話が複雑になる最大の理由は、これが行列の方程式である点である。そこで、行列のひとつの行と列に着目し、ベクトルとスカラーの最適化問題に帰着させ、行・列ごとに順繰りに解くことを考える。今、ある特定の変数 x_i に着目し、それが最後の行と列に来るように変数の名前を付け替えたと考え、$\mathsf{\Lambda}$ と $\mathsf{\Sigma}$ を、特に

$$\mathsf{\Lambda} = \begin{pmatrix} \mathsf{\Lambda}^{(-i)} & \boldsymbol{l} \\ \boldsymbol{l}^\top & \lambda \end{pmatrix}, \qquad \mathsf{\Sigma} = \begin{pmatrix} \mathsf{\Sigma}^{(-i)} & \boldsymbol{g} \\ \boldsymbol{g}^\top & \sigma \end{pmatrix} \tag{1.67}$$

のように分割して表したとしよう。$\mathsf{\Lambda}^{(-i)}$ と $\mathsf{\Sigma}^{(-i)}$ は、それぞれ $\mathsf{\Lambda}$ と $\mathsf{\Sigma}$ から x_i に対応する行と列を抜いた $(M-1)\times(M-1)$ 行列である。一方、$\boldsymbol{l}$ と $\boldsymbol{g}$ は $M-1$ 次元の列ベクトルになる。さらに計算の都合上、精度行列の逆行列を

$$\mathsf{W} \triangleq \mathsf{\Lambda}^{-1} = \begin{pmatrix} \mathsf{W}^{(-i)} & \boldsymbol{w} \\ \boldsymbol{w}^\top & w \end{pmatrix} \tag{1.68}$$

のように同様に分割しておく。$\mathsf{W\Lambda} = \mathsf{I}_M$ であるから、恒等式

$$\mathsf{W\Lambda} = \begin{pmatrix} \mathsf{W}^{(-i)}\mathsf{\Lambda}^{(-i)} + \boldsymbol{w}\boldsymbol{l}^\top & \mathsf{W}^{(-i)}\boldsymbol{l} + \boldsymbol{w}\lambda \\ \boldsymbol{w}^\top\mathsf{\Lambda}^{(-i)} + w\boldsymbol{l}^\top & \boldsymbol{w}^\top\boldsymbol{l} + w\lambda \end{pmatrix} = \begin{pmatrix} \mathsf{I}_{M-1} & \boldsymbol{0} \\ \boldsymbol{0}^\top & 1 \end{pmatrix} \tag{1.69}$$

が成り立つことに注意する。

我々の問題は、$\mathsf{\Sigma}$ に加えて $\mathsf{\Lambda}^{(-i)}$ と $\mathsf{W}^{(-i)}$ も与えられたと仮定した時に、$\boldsymbol{l}, \lambda, \boldsymbol{w}, w$ という 4 つの未知量を同時に求めるというものである。このような形で書けば、問題はベクトルとスカラーを未知量とする最適化問題になり、行

列を扱うよりも圧倒的に楽である。ただしそれと引き換えに、適切に解を初期化した上で、反復的に収束まで計算を繰り返すような算法とならざるを得ない。そのような方法が実用になるのか疑問に思うかもしれないが、graphical LASSO の提案当時[5] はもちろん、おそらく本書執筆時点においてもなお、これは計算速度と数値的安定性の双方において、精度行列を求めるための最も優れた算法と言われている。

対角要素 w の計算

上記の分割に基づいて、明示的に最適性条件 (1.66) を書き下してみる。

$$\begin{pmatrix} \mathsf{W}^{(-i)} & \boldsymbol{w} \\ \boldsymbol{w}^\top & w \end{pmatrix} = \begin{pmatrix} \Sigma^{(-i)} & \boldsymbol{g} \\ \boldsymbol{g}^\top & \sigma \end{pmatrix} + \rho \operatorname{sign} \begin{pmatrix} \Lambda^{(-i)} & \boldsymbol{l} \\ \boldsymbol{l}^\top & \lambda \end{pmatrix} \tag{1.70}$$

$\boldsymbol{l}, \lambda, \boldsymbol{w}, w$ という 4 つの未知量のうち、w の満たすべき式について考えてみよう。Λ は正定値であるため、その対角要素は正でなければならない[7]。したがって $\operatorname{sign}(\lambda) = 1$ で、最適性条件 (1.66) の右下の対角要素に関する部分について

$$w = \sigma + \rho \tag{1.71}$$

と書かれる。これで 4 つの未知量のうちまず w が求められた。

非対角要素 $\boldsymbol{w}$ の満たすべき式

次に非対角要素について考える。行列形式の最適性条件 (1.70) の右上部分は明らかに

$$\boldsymbol{w} - \boldsymbol{g} - \rho \operatorname{sign}(\boldsymbol{l}) = 0 \tag{1.72}$$

と書かれる。この式において、恒等式 (1.69) の右上部分を使って $\boldsymbol{l}$ を消去することを考えよう。$\boldsymbol{l} = -\lambda(\mathsf{W}^{(-i)})^{-1}\boldsymbol{w}$ となるから、上の条件は

$$\boldsymbol{w} - \boldsymbol{g} + \rho \operatorname{sign}\left((\mathsf{W}^{(-i)})^{-1}\boldsymbol{w}\right) = 0 \tag{1.73}$$

となることが分かる。ただし、Λ は正定であるからその対角要素である λ は正で、したがって符号関数の中ではあってもなくても関係ないことを用いた。ここで未知量 $\boldsymbol{w}$ の代わりに、新しい変数 $\boldsymbol{\beta} \triangleq (\mathsf{W}^{(-i)})^{-1}\boldsymbol{w}$ を導入しよう。すると上式は

[7] 証明は[21] の 10.4.2 節参照

$$\mathsf{W}^{(-i)}\boldsymbol{\beta} - \boldsymbol{g} + \rho\,\mathrm{sign}\,(\boldsymbol{\beta}) = 0 \tag{1.74}$$

と書かれる。この式は ℓ_1 正則化つき線形回帰において回帰係数を求める方程式とよく似ている。これは未知ベクトル $\boldsymbol{\beta}$ についての方程式である。第 l 成分を明示的に書くと次のようになる。

$$\sum_m W_{l,m}^{(-i)}\beta_m - g_l + \rho\,\mathrm{sign}(\beta_l) = 0 \tag{1.75}$$

最適性条件 (1.74) の解

この方程式に対する形式的な解は

$$\beta_l = \frac{1}{W_{l,l}^{(-i)}}[A_l - \rho\,\mathrm{sign}(\beta_l)] \tag{1.76}$$

で与えられる。ただし、

$$A_l \triangleq g_l - \sum_{m\neq l} W_{l,m}^{(-i)}\beta_m \tag{1.77}$$

と定義した。共分散行列の正定値性より $W_{l,l}^{(-i)} > 0$ が成り立つことに注意すると、式 (1.76) においては、A_l と $\pm\rho$ の大小関係により β_l の符号が左右されることがわかる。たとえば $A_l > \rho$ なら、右辺はどうやっても負にはならないので、$\beta_l > 0$ しかありえない。したがって、カッコの中身は $A_l - \rho$ しかありえない。$A_l < -\rho$ なら右辺はどうやっても正にはなれないので $\beta_l < 0$ であり、したがって、カッコの中身は $A_l + \rho$ しかありえない。

興味深いのは $|A_l| < \rho$ の場合である。この場合は、右辺は正にも負にもなりえる。右辺が正になるのはカッコの中身が $A_l + \rho$ となる場合であるが、これは $\mathrm{sign}(\beta_l) = -1$ のときに生ずる。しかしこれは右辺も左辺も正という仮定に反している。この場合、唯一の可能性は $\beta_l = 0$ となること（したがって $\mathrm{sign}(\beta_l)$ が何か ±1 以外の値をとること）しかない。右辺が負になるときも同様の矛盾を導けるので、$\beta_l = 0$ しかありえないことがわかる。これが先ほど「隠し玉」と表現した不定性の使い道である。結局、各 l に対して次のような解を得る。

$$\beta_l = \begin{cases} (A_l - \rho)/W_{l,l}^{(-i)}, & A_l > \rho \\ 0, & -\rho \leq A_l \leq \rho \\ (A_l + \rho)/W_{l,l}^{(-i)}, & A_l < -\rho \end{cases} \tag{1.78}$$

これより、$\rho > 0$ ならば、$\boldsymbol{\beta}$ は厳密な 0 を多く含んだ**疎**な、もしくは**スパース** (sparse) なベクトルになることが分かる。以下に見るとおり、$\boldsymbol{\beta} \propto \boldsymbol{l}$ が成り立つから、この結果として、Λ もまた疎な行列になる傾向がある。直感的には ρ は、精度行列の要素に付したしきい値のような役割を担う。ただし、$\rho = 0$ における精度行列の解 Σ^{-1} を（一般化逆行列なりで）何とか求め、その要素にしきい値を付し、絶対値がしきい値以下なら 0 にする、といった方法で疎な行列を作ると、一般には正定値などの条件を満たさず、ガウス分布としての首尾一貫性を失ってしまうというので注意が必要である。加えて、逆行列を明示的に求めるのは先に述べた通り数値計算的に難しい。

疎な精度行列を得るために（したがって重要な依存関係をうまく推定するために）ρ が重要な役割を担っているならば、どう ρ を決めるべきか、というのが実用上重要な問題となりえる。構造学習の文脈で、例えば、独立性の検定などの手法はいろいろと提案されているが、実用的に使えるものは乏しい。ひとつの合理的な方法は、graphical LASSO の枠外で、たとえば、求められた分布を異常検知に使うとして、異常検知の精度を最大にするように ρ を決める、というものである。

残る未知量 $\boldsymbol{l}, \lambda, \boldsymbol{w}$ の計算

さて、上記の結果からいよいよ、4 つの未知量 $\boldsymbol{l}, \lambda, \boldsymbol{w}, w$ のうち残る 3 つ、すなわち、$\boldsymbol{l}, \lambda, \boldsymbol{w}$ を求めてゆこう。まず $\boldsymbol{\beta}$ の定義式から

$$\boldsymbol{w} = \mathsf{W}^{(-i)}\boldsymbol{\beta} \tag{1.79}$$

が出る。残りのふたつ $\boldsymbol{l}, \lambda$ は、恒等式 (1.69) の右上部分と右下部分

$$\mathsf{W}^{(-i)}\boldsymbol{l} + \lambda\boldsymbol{w} = \boldsymbol{0} \tag{1.80}$$

$$\boldsymbol{w}^\top\boldsymbol{l} + w\lambda = 1 \tag{1.81}$$

を使って求めることができる。先に述べたように、前者から

$$\boldsymbol{l} = -\lambda(\mathsf{W}^{(-i)})^{-1}\boldsymbol{w} = -\lambda\boldsymbol{\beta} \tag{1.82}$$

となる。これを後者に代入して λ について解くと

$$\lambda = \frac{1}{w - \boldsymbol{w}^\top\boldsymbol{\beta}} = \frac{1}{w - \boldsymbol{\beta}^\top\mathsf{W}^{(-i)}\boldsymbol{\beta}} \tag{1.83}$$

Algorithm 3 Graphical LASSO

入力: $\mathsf{\Sigma}$ および ρ
初期化: $\mathsf{W} = \mathsf{\Sigma} + \rho \mathsf{I}_M$
repeat
　$1, \ldots, M$ から順に（またはランダムに）ひとつ i を選ぶ。
　$\mathsf{\Sigma}$ を式 (1.67) のように分割して $\boldsymbol{g}, \sigma$ を求める
　W を式 (1.68) のように分割して $\mathsf{W}^{(-i)}$ を求める。
　$w \leftarrow \sigma + \rho$
　$\boldsymbol{\beta} \leftarrow$ 式 (1.78)
　$\boldsymbol{w} \leftarrow \mathsf{W}^{(-i)}\boldsymbol{\beta}$
　$\lambda \leftarrow 1/(w - \boldsymbol{\beta}^\top \mathsf{W}^{(-i)}\boldsymbol{\beta})$
　$\boldsymbol{l} \leftarrow -\lambda\beta$
　$\boldsymbol{w}, w, \boldsymbol{l}, \lambda$ を使って $\mathsf{W}, \mathsf{\Lambda}$ を更新
until 収束
出力: W および $\mathsf{\Lambda}$

が得られる。これを式 (1.82) に入れ直すと

$$\boldsymbol{l} = -\frac{\boldsymbol{\beta}}{w - \boldsymbol{\beta}^\top \mathsf{W}^{(-i)}\boldsymbol{\beta}} \tag{1.84}$$

のように求められる。

以上は変数 i を狙ったと想定した時の結果である。したがって、最終的な解 $\mathsf{\Lambda}^*$、およびその副産物として得られる逆行列 W^* を得るためには

$$\begin{aligned}
w &\leftarrow \sigma + \rho \\
\boldsymbol{\beta} &\leftarrow \text{式 (1.78)} \\
\boldsymbol{w} &\leftarrow \mathsf{W}^{(-i)}\boldsymbol{\beta} \\
\lambda &\leftarrow \frac{1}{w - \boldsymbol{\beta}^\top \mathsf{W}^{(-i)}\boldsymbol{\beta}} \\
\boldsymbol{l} &\leftarrow -\frac{\boldsymbol{\beta}}{w - \boldsymbol{\beta}^\top \mathsf{W}^{(-i)}\boldsymbol{\beta}}
\end{aligned}$$

を $i = 1, 2, \ldots, M, 1, 2 \ldots$ と収束するまで何周でも繰り返す必要がある。これらの更新式の右辺には、$\mathsf{\Sigma}$ および W に由来する量しか出てこないことに注意されたい。全体の流れを Algorighm 3 にまとめておこう。

1.5.4　分散分権型学習問題の数値例

以上、かなり詳しく graphical LASSO の一般的な解法を解説した。ここで元の分散分権型の学習問題に移り、具体的な例題を考えてみることにしよう。

多様性のある複数のモデルの学習

参加者 $S = 3$ 人の図 1.7 に示すような系を考えよう。例えばそれぞれの参加者は同種の化学プラントの管理者で、何か $M = 4$ 個のセンサーの値を記録していると考えてもよい。これらのデータ源は、$K = 3$ のパターンを共有しており、図に示す通りの確率で観測するとする。たとえば、参加者 1 のプラントはパターン A を確率 $\frac{2}{3}$、パターン B を $\frac{1}{3}$ で観測する。すなわち、$\boldsymbol{\pi}_{\mathrm{A}} = (\frac{2}{3}, \frac{1}{3}, 0)^{\top}$ である。パターン A、B、C は、それぞれ異なる平均と共分散行列を持っている。すなわち

$$\boldsymbol{\mu}_{\mathrm{A}} = (5,0,0,5)^{\top}, \quad \boldsymbol{\mu}_{\mathrm{B}} = (0,5,5,0)^{\top}, \quad \boldsymbol{\mu}_{\mathrm{C}} = (0,0,0,0)^{\top} \tag{1.85}$$

と

$$\Lambda_{\mathrm{A}} = \begin{pmatrix} 1.2 & 0.0 & 0.0 & 1.0 \\ 0.0 & 1.2 & 1.0 & 0.0 \\ 0.0 & 1.0 & 1.2 & 0.0 \\ 1.0 & 0.0 & 0.0 & 1.2 \end{pmatrix}, \quad \Lambda_{\mathrm{B}} = \begin{pmatrix} 1.2 & 0.0 & 1.0 & 0.0 \\ 0.0 & 1.2 & 0.0 & 1.0 \\ 1.0 & 0.0 & 1.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 1.2 \end{pmatrix} \tag{1.86}$$

および

$$\Lambda_{\mathrm{C}} = \begin{pmatrix} 1.2 & 1.0 & 0.0 & 0.0 \\ 1.0 & 1.2 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.2 & 1.0 \\ 0.0 & 0.0 & 1.0 & 1.2 \end{pmatrix} \tag{1.87}$$

である。これを正解としてデータを無作為抽出で生成したとして、この正解モデルを再現できるか、というのが問題である。

複数の異なるパターンの混合モデルを学習する場合、パターンの数 K が事前にわからないという点が問題になり得る。しかし本章で説明した確率モデルの定式化に従えば、この点は大きな問題にならない。真の K を含む程度に十分大きな K_0 を仮定してパラメターの初期化を行い、$\gamma - 1$ を小さな正の数と

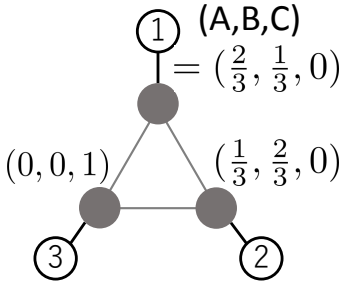

図 1.7　$S = 3, K = 3$ の分散分権型学習の例。各参加者は、指定された分布に従い、$M = 4$ 次元の標本を $N^s = 300$ 個無作為に生成する。

して、EM 反復を実行する。各反復の回において式 (1.58) の N_k を確認し、0 または 0 に非常に近い値になった時点でそのパターンをモデルから除去する。収束時に生き残るパターンの数が K の推定値ということになる。これは K_0 個の要素を持つ混合モデルをスパースに推定したとも言える。本節のモデルは、精度行列におけるスパース性と、混合要素にわたるスパース性という二つのスパース性を持っていることになる。

なお、このような手法で $K < K_0$ を見積もる方法は、ベイズ学習の**関連度自動決定** (automatic relevance determination) の文脈で以前から知られていた[1],[2] が、それはどちらかといえば経験則に近かった。紙幅の都合上ここでは触れないが、最近、ℓ_0 正則化理論を援用して、数学的に厳密にこのクラスタ数選択機能を定式化する方法も提案され[14]、理論的にも十分信頼できる土台を備えた方法になっていることを付言しておく。

図 1.8 に、このデータからのモデル推定結果を示す。図では $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \boldsymbol{\pi}_3$ を棒グラフとして示してある。パターン数自動決定機能を強調するため、初期化時点での $K_0 = 6$ 個のインデックスの上に確率値を示している。図から分かる通り、$K_0 = 6$ がら出発した冗長なモデルが、$K = 3$ まで適切に刈り込まれたことが分かる。計算された確率値も図 1.7 に示した正解の値をほぼ完全に再現していることが分かる。

共同パターン辞書学習法でモデルパラメター $\{\boldsymbol{\pi}^s\}_{s=1}^S$ および $\{\boldsymbol{\mu}_k, \Lambda_k\}_{k=1}^K$

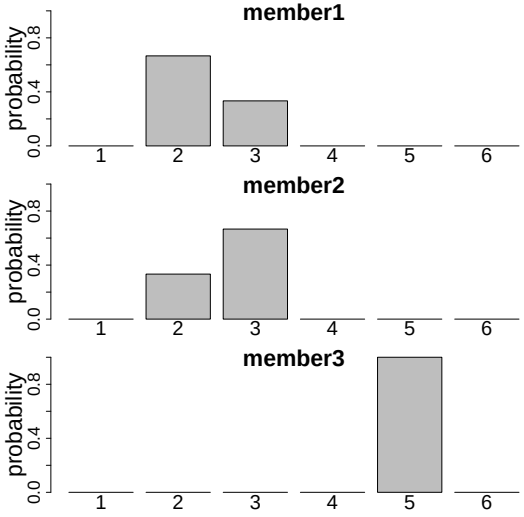

図 1.8 パターン重み $\{\boldsymbol{\pi}^s\}_{s=1}^S$ についての収束解。参加者ごとに異なる分布が学習されていることが分かる。これは $K_0 = 6$ を前提にランダムに初期化した状態から出発して求められたものである。

が求められたら、参加者 s の測定値についての分布が

$$p(\boldsymbol{x}^s \mid \boldsymbol{\Theta}, \boldsymbol{\Pi}) = \sum_{k=1}^{K} \pi^s \mathcal{N}(\boldsymbol{x}^s \mid \boldsymbol{\mu}_k, \Lambda_k^{-1}) \tag{1.88}$$

のように求められる。ひとつ興味ある量は、今観測したデータ $\boldsymbol{x}^{s'}$ が異常かどうかというものである。第 1 章で述べた通り、対数損失

$$a(\boldsymbol{x}^{s'}) = -\ln p(\boldsymbol{x}^{s'} \mid \boldsymbol{\Theta}, \boldsymbol{\Pi}) \tag{1.89}$$

がそのための異常度の指標になりえる。さらに進んで、変数ごとの異常度を計算したい場合もありえる。変数同士の相関を無視せずにこれを行うのは自明な問題ではないが、ガウス型グラフィカルモデルの枠内でマルコフ確率場とみなし条件付き確率を計算する方法[8] が知られている。また、M 個の変数の中で

いくつかを出力変数として残りを入力とみなせるような場合は、モデルによらずに使える汎用的な異常寄与度の計算方法も利用できる[7]。

暗号化計算との計算時間の比較

上の計算例では、動的合意法は統計量 $\{\bar{N}_k, \bar{\boldsymbol{m}}_k, \bar{\mathsf{C}}_k\}_{k=1}^K$ の中に非明示的に組み込まれており、結果に反映されることはなかった。最後にこの点について見てみよう。

合意形成において、無作為分割法のような確率的保証しかつけられない「乱暴な」方法ではなく、確実な暗号学的安全性の保証をつけたいという場合もあるはずである。信頼できる中央のサーバーの存在を仮定できれば、単純にサーバーとの通信を暗号化する問題であるが、分散分権型の設定では問題が格段に複雑になる。この点について、準同型暗号を使った合意形成手法が最近提案されている[16]。これは動的合意法における式 (1.27) の更新を、暗号化された情報を復号化することなく計算するプロトコルである。

ここでは、準同型暗号に基づく手法と無作為分冊法に基づく手法の間で、単発の合意形成手続きにおいてどれだけ計算時間が異なるのかについての定性的な比較を行う。通信路としては $b = 2$ のサイクルグラフを採用し、図 1.4 と同様に、$[-10, 10]$ の一様乱数で各参加者の持っている初期値 $\xi^s(0)$ を初期化した。異なる S ごとに、2 乗誤差 (root mean-squared error; RMSE) が 0.01 になったところで収束とした。無作為分割法では $N_{\mathrm{C}} = 5$ とし、準同型暗号の計算は Paillier 暗号を実装した R のライブラリ `homomorpheR`[13] を用いた。

計算時間の比較を図 1.9 に示す。無作為分割法は、回数にして $N_{\mathrm{C}} = 5$ 倍の合意形成を行っているにも関わらず、計算時間が数桁速いことが分かる。準同型暗号に基づく安全な連合学習の算法の開発は最近の機械学習応用の主要な課題のひとつになっている。この実験結果において、計算時間の隘路となる部分のひとつは鍵の生成にある。この点、たとえば鍵を再利用したり、暗号化に特化した計算手段を使ったり、といった工夫の余地は大いにある。

ここで重要なことは、用途により異なる計算手法があってもよいということである。たとえば個人情報を含むような重要なカテゴリカルデータと、工場の生産設備の出すノイズの乗った測定値とでは、仮に両者とも機密情報であったとしても、その扱いは違っていて当然である。前者と違い後者のデータでは、密度推定や時系列予測など、何かある程度高度な統計処理が必要とされること

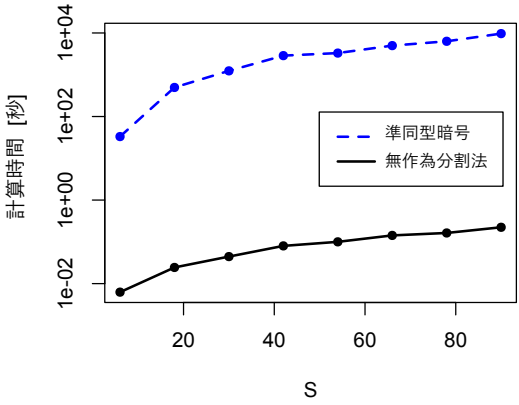

図 1.9　準同型暗号に基づく合意形成算法[16] と、無作為分割法との計算時間の比較。

が多い。その場合、ビットコイン流の、本章で見たような確率的決着性を持つ算法が必要になる場面もあるかもしれない。

1.6　ま　と　め

本章では分散分権型の設定での機械学習アルゴリズムの構成法について議論した。分散分権型の機械学習は、民主主義、多様性、プライバシーという 3 つの制約条件を持つ学習の問題である。本質的にはそれは分散型のマルチタスク学習とみなせるが、中央のサーバーの存在を前提としない場合、これまで考えられてきたものとは異なる技術的問題が生ずる。個々の課題は一般に、想定される機械学習のモデルに依存するが、指数型分布族については、参加者間の協業は秘匿集計という比較的単純な数学的操作に帰着される。データプライバシーを保ちつつそれを行うことは簡単ではないが、ブロックチェーン技術の本質を、無作為性に基づくセキュリティ「保証」の技術と位置付けることで、新たな視界が開けてくる。準同型暗号を使った安全な合意形成[16] や秘密分散[17] などに基づく厳格な方法に代えた軽量な選択肢として、無作為分割法という手法を説明した。

最後に、いくつかの残る課題について述べておこう。ひとつは、中央集権的サーバーと契約による法律的強制力を組み合わせた従来の管理モデルとの得失をより深く把握することである。この点は、通常の（パブリック型の）ブロックチェーンとプライベート型のブロックチェーンの間の比較においてよく話題になる。プライベート型の、とりわけ「パーミッション型」と呼ばれる会員資格審査を経た上での小規模な協業環境においては、あえて分権型を選択する実用面での理由は乏しい。パーミッション（許可）を出す権威者がいてもいいのなら、中央のサーバー管理者がいても悪くないだろう、という話になるからである。問題となるのは、国際協業が発生するような場面である。法的強制力は国境の壁を超えることはできない。したがって、設計上プライバシーと協業による便益の双方が保証される学習方法には意義がある。その場合であっても、本文中で触れたメタ合意の問題など、実運用上に解決すべき問題はまだ多い。

第 2 に、通信におけるネットワーク障害の可能性を考えに入れることである。これは大きく分けて通信路の切断と遅延という 2 つの側面がある。ネットワーク障害が特定のリンクで起こるのか、無作為に起こるのかによってもそのインパクトは異なる。ネットワーク障害に頑強な算法を構築しつつ、かつ、望ましい状態からのずれを定量的に評価するような手法の開発が必要となろう。

第 3 に、これはブロックチェーンそのものにも当てはまるが、従来の暗号学的な方法との得失をより深く理解することが必要である。我々の経験では、たとえば準同型暗号を組み込んだ EM 学習はそうでない場合（たとえば無作為分割法）とくらべて数桁という程度で計算時間が多く、規模拡張性に難がある。しかし一方で、確率的にしか安全性を言えない算法がビジネス的に受け入れられがたいのも確かである。おそらく両者の良い面を組み合わせたような仕組みが必要であろう。

第 4 に、上記の点とも関係するが、プライバシー漏洩についての定量的評価技術を確立することである。これまで差分プライバシーなどいくつかのプライバシー評価法が知られているが、たとえば Minami ら[11] により指摘された通り、本章で想定したような非カテゴリカルデータに対しては、従来の評価手法は必ずしも満足できるものではない。協業学習のビジネス応用を考えた場合も、例えばモデルの外販に際し、どういうプライバシーリスクが考えられるのかという点は明示する必要がある。また、共謀や盗聴など個々の主要な攻撃シ

ナリオについても、ある程度明確な解答を与える必要がある。この点について、および関連するその他の論点について、我々の最近の論文[9] でやや詳しく論じているので参照されたい。

本章の内容は IBM 東京基礎研究所の Rudy Raymond 博士との共同研究に基づく。特に、無作為分割法と修正サイクルグラフの着想は Raymond 博士による。

参考文献

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

[2] Adrian Corduneanu and Christopher M Bishop. Variational Bayesian model selection for mixture distributions. In *Artificial intelligence and Statistics*, volume 2001, pages 27–34, 2001.

[3] Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *International conference on financial cryptography and data security*, pages 436–454. Springer, 2014.

[4] Shai Fine and Katya Scheinberg. Efficient svm training using low-rank kernel representations. *Journal of Machine Learning Research*, 2(Dec):243–264, 2001.

[5] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[6] Tsuyoshi Idé. Collaborative anomaly detection on blockchain from noisy sensor data. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 120–127. IEEE, 2018.

[7] Tsuyoshi Idé, Amit Dhurandhar, Jirı Navrátil, Moninder Singh, and Naoki Abe. Anomaly attribution with likelihood compensation. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 21)*, pages 4131–4138, 2021.

[8] Tsuyoshi Idé, Ankush Khandelwal, and Jayant Kalagnanam. Sparse Gaussian markov random field mixtures for anomaly detection. In *Proceedings of the 2016 IEEE International Conference on Data Mining (ICDM 16)*, pages 177–186, 2016.

[9] Tsuyoshi Idé and Rudy Raymond. Decentralized collaborative learning with

probabilistic data protection. In *Proceedings of the 2021 IEEE International Conference on Smart Data Services (SMDS 21)*, pages 234–243, 2021.

[10] Tsuyoshi Idé, Rudy Raymond, and Dzung T Phan. Efficient protocol for collaborative dictionary learning in decentralized networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI 19)*, pages 2585–2591, 2019.

[11] Kentaro Minami, HItomi Arai, Issei Sato, and Hiroshi Nakagawa. Differential privacy without sensitivity. In *Advances in Neural Information Processing Systems*, pages 956–964, 2016.

[12] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[13] Balasubramanian Narasimhan. homomorpheR. In *CRAN*. 2019.

[14] Dzung T Phan and Tsuyoshi Idé. ℓ_0-regularized sparsity for probabilistic mixture models. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 172–180. SIAM, 2019.

[15] Wei Ren, Randal W Beard, and Ella M Atkins. A survey of consensus problems in multi-agent coordination. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 1859–1864. IEEE, 2005.

[16] Minghao Ruan, Muaz Ahmad, and Yongqiang Wang. Secure and privacy-preserving average consensus. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and Privacy*, pages 123–129. ACM, 2017.

[17] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[18] Salil P Vadhan et al. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1–3):1–336, 2012.

[19] Yang Xiao, Ning Zhang, Wenjing Lou, and Y Thomas Hou. A survey of distributed consensus protocols for blockchain networks. *IEEE Communications Surveys & Tutorials*, 22(2):1432–1465, 2020.

[20] ギルバート ストラング. **線形代数とその応用**. 産業図書, 1978.

[21] 井手剛. **異常検知と変化検知**. 講談社, 2015.

[22] 井手剛. **入門 機械学習による異常検知** *–R* **による実践ガイド–**. コロナ社, 2015.

[23] 佐古和恵. 不正を防ぐ合意形成ルール「プルーフ・オブ・ワーク」. In **ブロックチェーン技術の未解決問題**, **第** *2* **章**. 日経 BP, 2018.

[24] 松尾真一郎, 楠正憲, 崎村夏彦, 佐古和恵, 佐藤雅史, and 林達也. **ブロックチェーン技術の未解決問題**. 日経 BP, 2018.

[25] 大石哲之. **ビットコインはどのようにして動いているのか？—ビザンチン将軍問題、ハッシュ関数、ブロックチェーン、*PoW* プロトコル**. tyk publishing, 2014.

[26] 藤澤洋徳 and 井手剛. **大規模計算時代の統計推論*:* 原理と発展**. 共立出版, 2020.

[27] 野口悠紀雄. **ブロックチェーン革命**. 日本経済新聞出版社, 2017.

索　引