Vol. 37 No. 4 情報処理学会論文誌 Apr. 1996

ニューラルコンピューティングによる小選挙区区割り手法

斎藤孝之[†] 武藤佳恭[††]

1994年12月に衆議院の小選挙区比例代表並立制導入に伴う全国300小選挙区の区割り法が成立した．この区割りの作業には，「人口の格差を2倍以内におさえる」「飛び地にしない」などのいくつかの条件がある．条件が増えれば増えるほど複雑さが増して作成が困難になる．これを電卓片手に人手で行うとなると大変な労力となるが，実際の選挙区作成は手作業で行われている．この論文では，ニューラルネットワークを用いたコンピュータによる選挙区の自動生成の手法を紹介する．ここでは，東京都の選挙区の作成を試みた．いくつかの条件を考慮して作成したところ，人口格差の点で現在の区割り案（1.47倍）よりもよい結果を（1.28倍）生成できた．

A Neural Computing Approach to the Single-member Electoral Constituency Districting System in Japan

TAKAYUKI SAITO[†] and YOSIYASU TAKEFUJI[††]

In December 1994, Japanese minor electorate system (single-member constituency system) for the House of Representatives was established where Japan is divided into three hundred constituencies. A single representative will be elected from each constituency. Zoning three hundred constituency was accomplished by hand calculators in Japan, although the constituency zoning is a very elaborate task because several constraints must be satisfied. This paper presents a neural computing approach for automatically zoning constituencies. Our method was examined by using 25 Tokyo constituencies. Ideally, the weight of a single vote in a certain population to elect a representative should be equal to that of the other constituencies. Based on the established rule, the ratio of the lightest weight to the heaviest weight must be within two. Our result shows that our ratio is 1.28 while the current (official) ratio is 1.47.

1. はじめに

1994年12月に衆院小選挙区区割り法案が国会を通過したことにより，これ以降の衆議院議員選挙は中選挙区制から小選挙区制と比例代表制で行われることになった．小選挙区区割り案は衆院議員選挙区画定審議会（石川忠雄会長）により1994年8月村山首相に勧告されたものである．選挙区の作成は人口格差など様々な条件を考慮して行われるため，大変複雑な組合せの問題である．実際の選挙区作成は自治省の担当役員によって手作業で行われている．そのため膨大な時間と労力がかかる．

本論文では短い時間でより良い区割りを行うための手法を紹介する．この手法はニューラルネットワークを用いコンピュータで選挙区を自動生成するというものである．ニューラルネットワークを用いる理由は2つある．1つはニューラルネットワークは複雑多様な条件を同時に扱えるので今回のように様々な制約条件を扱わなければならない問題の処理に効果的である，ということである．今回のシミュレーションにおいては選挙区作成の制約条件として，「人口の格差をできるだけ小さくする」「飛び地はなくす」等を取り上げた．実際の選挙区作成ではこのほかにも以前の中選挙区の区域を尊重する，地域事情を考慮するなどの制約条件が加わる．ニューラルネットワークでは制約条件を動作式あるいはデータで表現すればよいので，今回取り上げたもの以外でも対応可能である．新しい条件をつけ加える際は，データを追加・変更したり動作式に新しい項をつけ加えるだけでよいので，システムやシミュレーションの手続きを変更する必要はない．制約条件をもとにニューラル表現と動作式を作成すれば解はシステムが見つけだしてくれる．2つめの理由は，ニューラルネットワークはヒューリスティックな探索

† 慶應義塾大学大学院政策・メディア研究科
Graduate School of Media and Governance, Keio University
†† 慶應義塾大学環境情報学部
Faculty of Environmental Information, Keio University

を行うので高速に解を見つけだすことが可能である，ということである．今回のシミュレーションでも数秒で，遅いときでも数十秒で解を探し出せた．手作業で作成することを考えるとずっと速い．全探索ではないので出てきた解が必ずしも最適解とは限らないが，制約条件を満たした解を高速に見つけだすことができる．

今回のシミュレーションでは東京都の区割りを行った．人口格差をできるだけ小さくすることに重点を置いたところ，現在の小選挙区のものよりも格差が小さい区割りを行うことができた．その結果もあわせて紹介する．

2. ニューラルネットワークの原理

まずニューラルネットワークについて述べる．

ニューラルネットワークは，2つの構成要素からなる．1つは処理演算子であるニューロンと，もう1つは重み付シナプス・リンクである．1つのニューロンからの出力はシナプス・リンクを通じて他のニューロンに伝わる．ニューロンへの入力はそのときの結合の仕方によって変わる．

今回用いるニューラルコンピューティングの手法は，組合せ最適化問題を解く際に用いられるものである．最適化問題のためのニューラルネットワークの手法は，Hopfield,Tank 等[1]によって提案されたものをはじめ，武藤[2]や Szu 等[4]（その他，参考文献 5)）によってもいくつか提案されている．今回の選挙区の作成に関しては，武藤による手法を用いた．この手法では，あらかじめシナプス・リンクの接続を固定したニューラルネットワークを用いる．与えられた問題を解くにあたってニューラル表現を決定することはニューロン間のシナプス・リンクの重みを決定することに等しい．そしてシナプス・リンクの接続は与えられた問題の制約条件によって決まる．ニューラル表現は一般にニューロン動作式で与えられる．

ニューロン i の出力 V_i は入力 U_i の関数である．

$$V_i = f(U_i) \tag{1}$$

ここで入出力関数 f は非線形増加関数である．この入出力関数にはいくつかのモデルがあるが，今回用いる関数のモデルはヒステリシス・マッカロック-ピッツ（hysteresis McCulloch-Pitts）ニューロン入出力関数[4),8)]と呼ばれるもので，以下の式で表される．

$$\begin{aligned} V_i &= 1 \quad \text{if } U_i > UTP \text{ (Upper Trip Point)} \\ &= 0 \quad \text{if } U_i < LTP \text{ (Lower Trip Point)} \\ &\text{unchanged otherwise.} \end{aligned} \tag{2}$$

つねに $UTP > LTP$ である．これをグラフにしたのが**図 1** である．

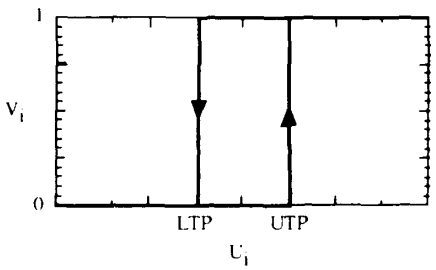

図 1 ヒステリシス・マッカロック-ピッツニューロン入出力関数
Fig. 1 Hysteresis McCulloch-Pitts neuron model.

ニューロンの一般動作式（motion equation）[4]は次式で表現される．

$$\frac{dU_i}{dt} = -\frac{\partial E(V_1, V_2, \ldots, V_n)}{\partial V_i} \tag{3}$$

ここで E はエネルギー関数と呼ばれている．最終的な目標は問題の制約条件を満たしながら，このエネルギー関数を最小化することにある（f が非線形増加関数の場合，エネルギー関数 E がつねに減少か増加しないことの証明は参考文献[4]を参照のこと）．エネルギーが大きいということは，それだけ制約条件を満たしていないということである．したがって，今回の問題のような組合せ最適化問題を解くにあたっては，エネルギー関数 E を減少させる正しい動作式を導き出すことが最も重要である．

3. 選挙区作成のシミュレーション

ニューラルネットワークを用いた選挙区作成のシミュレーションについて記す．

3.1 東京都の選挙区について

東京都の選挙区についてまとめると以下のようになる（実際の選挙区については**図 2**，**表 1** 参照）．

- 選挙区数 25.
- 区部は 17 選挙区，市部は 8 選挙区に分けられる．区部内の区と市が混ざった選挙区はない．
- 行政区数は区部が 29，市部が 28.
- 人口格差は約 1.47 倍.

区部の行政区数は 29 である．本来は 23 であるのだが，小選挙区制度では全国の各選挙区の人口は上限が 549382 人となっているが，区部内ですでにこの上限を上回っているために分割された行政区域が大田区・世田谷区・練馬区・足立区・江戸川区の合計 5 つあり，これに島部を 1 つの行政区として扱っているためである．

今回は実際の小選挙区と同じ行政区を用い，区部と市部を分けてシミュレーションを行った．

表1 東京都選挙区表
Table 1 Official constituencies of Tokyo.

選挙区	行政区	人口（人）	選挙区	構成行政区名	人口（人）
1	千代田区 港区 新宿区	494761	18	武蔵野市 三鷹市 小金井市	410540
2	中央区 文京区 台東区	412279	19	小平市 国分寺市 国立市 田無市 保谷市	501118
3	品川区 大田区 A 島部	539536	20	東村山市 東大和市 清瀬市 東久留米市 武蔵村山市	456053
4	大田区 B	485325	21	立川市 昭島市 日野市	424124
5	目黒区 世田谷区 A	506121	22	府中市 調布市 狛江市 稲城市	539897
6	世田谷区 B	534152	23	町田市 多摩市	493539
7	渋谷区 中野区	525312	24	八王子市	466347
8	杉並区	529485	25	青梅市 福生市 秋川市 羽村市 瑞穂町 西多摩郡	368036
9	練馬区 B	474650			
10	豊島区 練馬区 A	405883			
11	板橋区	518943			
12	北区 足立区 A	497455			
13	足立区 B	488355			
14	墨田区 荒川区	407753			
15	江東区	385159			
16	江戸川区 B	472633			
17	葛飾区 江戸川区 A	518107			

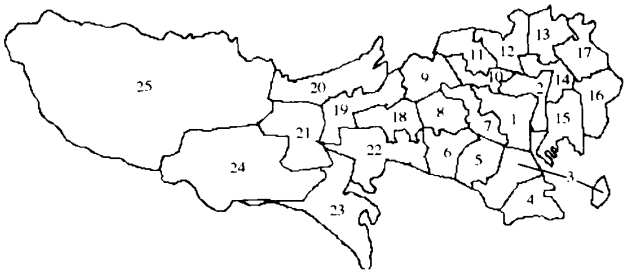

図2 東京都選挙区図
Fig. 2 The official constituency map of Tokyo (corresponding to Table 1).

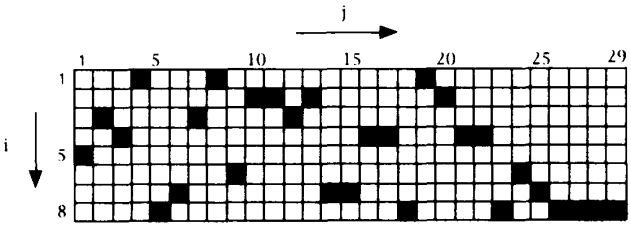

図3 ニューラル表現（市部）
Fig. 3 The neural representation for the rural part.

3.2 ニューラル表現

図3がニューラル表現である．$i \times j$ のニューロンの2次元配列であるが，i は選挙区を j は行政区をそれぞれ表している．図3は市部のためのニューラル表現である．市部は行政区が28あるが，西多摩郡の中の瑞穂町が地理的に離れているため別に扱った．そのため市部では8×29のニューロンの配列を用意する．区部は17×29のニューロンの配列を用意する．i 行 j 列のニューロンの入力を U_{ij} で，出力を V_{ij} でそれぞれ表す．図3でマス目が黒くなっているところはニューロンが発火（出力 V_{ij} が1になること）していることを示す．これは j 行政区が i 選挙区に属していることを意味する．マス目が白い部分は j 行政区が i 選挙区に属していないことを意味する．

3.3 必要なデータ

シミュレーションに必要なデータは以下の2つである．

- 人口データ
- 隣接データ

「人口データ」は平成2年国勢調査のものである．各新聞報道で用いられていたデータであり，実際の区割りと比較するために同じデータを用いた．「隣接データ」とは1つの区や市が他のどの区または市と接しているかのデータである．これは今回のシミュレーションにおける「飛び地をなくす」という制約条件を満たすために必要となるデータである．なお，分割された行政区に関しては名前の後にAとBをつけて区別した．Aは人口が少ない方，Bは人口の多い方である（**表2**参照）．

3.4 制約条件

シミュレーションにおける区割りの制約条件を以下に記す．

(1) 各行政区は1つの選挙区に属する．
(2) 選挙区の人口の上限…540000人，
　下限…400000人．
(3) 選挙区は飛び地にしない．
(4) 同じ選挙区に属するのは区部で2つ隣まで市部で3つ隣まで．

(1)の条件は図3のニューラル表現でいえば，各列に発火しているニューロンの数が1個でなければなら

表 2 人口・隣接データ
Table 2 The population and adjacency data.

行政区名	隣接行政区	人口(人)	行政区名	隣接行政区	人口(人)
千代田区	中央 新宿 文京 港 台東	39472	八王子市	昭島 多摩 町田 秋川 西多摩郡	466347
中央区	千代田 港 台東 墨田 江東 島部	68041	立川市	小平 国分寺 国立 日野 昭島 福生 武蔵村山 東大和	152824
港区	千代田 中央 新宿 渋谷 品川 島部	158499	武蔵野市	保谷 田無 小金井 三鷹	139077
新宿区	千代田 文京 港 豊島 中野 渋谷	296790	三鷹市	武蔵野 小金井 府中 調布	165564
文京区	千代田 新宿 豊島 北 台東	181269	青梅市	羽村 秋川 瑞穂 西多摩郡	125960
台東区	千代田 中央 文京 荒川 墨田	162969	府中市	国立 国分寺 小金井 三鷹 調布 稲城 多摩 日野	209396
墨田区	中央 江東 台東 荒川 葛飾 江戸川B 足立B	222944	昭島市	八王子 福生 立川 日野	105372
江東区	中央 墨田 江戸川B 島部	385159	調布市	三鷹 府中 狛江	197677
品川区	港 目黒 大田A 大田B 島部	344611	町田市	八王子 多摩	349050
目黒区	渋谷 品川 世田谷A 世田谷B 大田A	251222	小金井市	武蔵野 三鷹 府中 国分寺 小平	105899
大田区A	目黒 品川 大田B 世田谷A	162589	小平市	東大和 東村山 東久留米 田無 小金井 国分寺 立川	164013
世田谷区A	目黒 大田A 世田谷B	254899	日野市	八王子 昭島 立川 国立 多摩 府中	165928
渋谷区	新宿 港 中野 杉並 世田谷B 目黒	205625	東村山市	小平 東大和 清瀬 東久留米	134002
中野区	新宿 渋谷 杉並 練馬A 練馬B	319687	国分寺市	小平 小金井 府中 国立 立川	100982
杉並区	中野 渋谷 世田谷B 練馬B	529485	国立市	国分寺 府中 日野 立川	65833
豊島区	文京 新宿 北 板橋 練馬A	261870	田無市	武蔵野 小平 保谷 東久留米	75144
北区	文京 豊島 荒川 板橋 足立A	354647	保谷市	武蔵野 田無 東久留米	95146
荒川区	台東 墨田 北 足立A 足立B	184809	福生市	立川 昭島 秋川 羽村 瑞穂 武蔵村山	58062
板橋区	練馬A 練馬B 豊島 北	518943	狛江市	調布	74189
練馬区A	豊島 中野 板橋 練馬B	144013	東大和市	立川 武蔵村山 小平 東村山	75132
足立区A	北 荒川 足立B	142808	清瀬市	東村山 東久留米	67539
葛飾区	墨田 足立B 江戸川A 江戸川B	424801	東久留米市	保谷 小平 東村山 清瀬 田無	113818
江戸川区A	葛飾 江戸川B	93306	武蔵村山市	立川 東大和 福生 瑞穂	65562
大田区B	品川 大田A 島部	485325	多摩市	八王子 府中 稲城 町田 日野	144489
世田谷区B	渋谷 目黒 杉並 世田谷A 島部	534152	稲城市	多摩 府中	58635
足立区B	墨田 荒川 葛飾 足立A	488355	秋川市	八王子 青梅 福生 羽村 西多摩郡	50387
練馬区B	中野 杉並 板橋 練馬A	474650	羽村市	青梅 福生 秋川 瑞穂	52103
江戸川区B	葛飾 墨田 江東 江戸川A 島部	472633	瑞穂町	青梅 福生 武蔵村山 羽村	30967
島部	中央 港 品川 江東 大田B 江戸川B	32336	西多摩郡	八王子 青梅 秋川	50557

ないということである．(2) は人口の制約条件である．実際の小選挙区制度では人口の上限は 549382 人・下限は 274692 人となっているが，今回のシミュレーションは少しでも定数の格差を縮める区割りを行いたいと考えたので独自に定めた．表 2 を見て分かるように，世田谷区 B の人口がすでに 534152 人であるので上限は 540000 人に定めた．上限をこれ以上下げることができないので，実際の区割りよりも格差を縮めるためには下限を上げなければならない．そこで適当な値として 400000 人に定めた．この条件を満たせば人口格差 1.35 倍以下の選挙区を作成できることになる．これらの値は区分けを行う地域の人口によって変わる．(3) は実際の区割り作業の際にも考慮されているものである．(4) は独自に加えたもので，なるべく近い行政区どうしが同じ選挙区になるようにするもの．人口の制約を満たすからといって縦あるいは横に長くつながっているなどの現実的ではない選挙区を作成するよりも，できるだけコンパクトにまとまった選挙区を作成する方がよいと考えた．

3.5 動 作 式

3.4 節の制約条件をもとに作成された動作式が以下の式 (4) である．

$$\frac{dU_{ij}}{dt} = -\left(\sum_{k=1}^{n} V_{kj} - 1\right) - f_1(i,j) - \left(\sum_{\substack{k=1 \\ k\neq j}}^{m} f_2(i,j,k) V_{ik} V_{ij}\right). \tag{4}$$

ここで，f_1 および f_2 は以下のとおり．

$$\begin{aligned} f_1(i,j) &= N \quad \text{if} \quad \sum_{k=1}^{m} V_{ik} P_k > UL \\ &= -N \quad \text{if} \quad \sum_{k=1}^{m} V_{ik} P_k < LL \\ &= 0 \quad \text{otherwise.} \end{aligned} \tag{5}$$

$$
\begin{aligned}
f_2(i,j,k) &= 0 \quad \text{if} \quad D_{jk} = 0 \\
&= 0 \quad \text{if} \quad D_{jp} = D_{kp} = 0 \\
&\quad \text{and} \quad V_{ip} = 1 (1 \le p \le m, p \ne j) \\
&= 0 \quad \text{if} \quad D_{jp} = D_{pq} = D_{qk} = 0 \\
&\quad \text{and} \quad V_{ip} = V_{iq} = 1 (1 \le p \le m, \\
&\qquad 1 \le q \le m, p \ne j, q \ne j, p \ne q) \\
&= 1 \quad \text{otherwise.} \qquad (6)
\end{aligned}
$$

式(4)の第1項は制約条件(1)を満たすためのもので，j 列には1個のニューロンが発火するようにしている．n は選挙区数で，区部の場合 $n = 17$，市部の場合 $n = 8$.

第2項は制約条件(2)の人口の制約条件を満たすためのものである．関数 f_1 において，P_x は行政区 x の人口を表し，UL および LL は上限（Upper Limit）と下限（Lower Limit）をそれぞれ表している．今回のシミュレーションでは $UL = 540000 \cdot LL = 400000$ である．N は定数であまり小さい値だと収束しなくなる．シミュレーションの際に適当な値を設定する．区部のシミュレーションでは $N = 7$，市部では $N = 5$ のときに解によく収束するようになったので，これらの値を用いた．m は行政区数で区部・市部ともに $m = 29$．この項は，i 選挙区において人口が上限を越えているとニューロンの発火を抑え，下限を下回っているとニューロンの発火をうながし，それ以外のときは条件を満たしているのでニューロンの状態に影響を与えない．

第3項は制約条件(3)，(4)を満たすための項である．関数 f_2 において D_{xy} は隣接データで，行政区 x が y と接しているまたは $x = y$ のときは $D_{xy} = 0$，接していないときは $D_{xy} = 1$．1番目の if では行政区 j が k と直接接している場合，2番目が k が2つ隣の場合，3番目が k が3つ隣の場合である．ただし，区部の場合は2つ隣までなので3番目の if は適用しない．それぞれの条件のいずれかを満たしていると，i 選挙区において j と k は飛び地ではなくかつ同じ選挙区に属する範囲条件以内ということになる．この項では，i 選挙区において j 行政区および k が発火状態であってかつ j と k が飛び地である場合にはニューロンの発火を抑制し，それ以外のときはニューロンの状態をそのままにしておく．

式(4)に2つの項を付け足す必要がある．これら2つの項をつけ加えることにより，収束までの時間が短縮され，初期値を変えても収束するようになった．

まず1つはヒルクライミング（hill-climbing）項[2),4)]と呼ばれるもので，極小解（local minimum）の状態から抜け出し，すべての制約条件を満たす解へできるだけ導くためのものである．

$$
-Ch\left(\sum_{k=1}^{n} V_{kj}\right). \qquad (7)
$$

ここで，

$$
\begin{aligned}
h(X) &= 2 \quad \text{if} \quad X = 0 \\
&= 0 \quad \text{otherwise.} \qquad (8) \\
C &= 4 \quad \text{if} \quad t \bmod 20 < 5 \\
&= 1 \quad \text{otherwise.} \qquad (9)
\end{aligned}
$$

C はヒルクライミング項に強弱をつけるもので，シミュレーションでは式(9)のように設定した（ここで "$t \bmod 20$" は t を20で割った余り）．t は繰返し数（iteration steps）であり，$i \times j$ 個のニューロンを1度ずつ計算した後 t を更新して $t = t + 1$ とする．シミュレーションを行っていると，ある列に1つのニューロンも発火しないまま状態が変化しなくなってしまうときがある．一種の極小解の状態である．この状態のときに強制的にその列のニューロンが発火させて，極小解から抜け出す働きをするのがヒルクライミング項である．極小解の状態によっては2を加えた程度では抜け出せない場合があるので，C を用いて強弱をつけることで極小解から抜け出す働きを強める．ただし，C の値が大きすぎたり，頻繁に強めたりするとかえって収束の妨げとなる．ヒルクライミング項や C にどのような値を設定するかは解決する問題による．シミュレーションを通じて適度な値を設定する．今回の場合，式(8)，(9)のように設定したとき解への収束が良くなったのでこれらの値を用いた．

もう1つの項は以下の式(10)である．

$$
-C\left(\sum_{k=1}^{m} V_{ik} D_{jk}\right) f_3(i)/M. \qquad (10)
$$

ここで f_3 は以下のとおり．

$$
\begin{aligned}
f_3(i) = 1 \quad \text{if} \quad & \sum_{y}^{m} \left(-\left(\sum_{k=1}^{m} V_{ky} - 1 \right) - f_1(i,y) \right. \\
& \left. - \left(\sum_{k=1, k \ne y}^{m} f_2(i,y,k) V_{ik} V_{iy} \right) \right) \ne 0 \\
= 0 \quad & \text{otherwise.} \qquad (11)
\end{aligned}
$$

この項では，i 選挙区において j 行政区と直接隣接していない行政区（2つ以上隣の行政区）の発火数を計算している．東京都の場合，特に区部では1選挙区に含まれる行政区数が少ないので，i 選挙区において j 行政区と直接隣接していない行政区が多く発火している場合，j 行政区は i 選挙区に所属する可能性は少なくなる．また，制約条件(4)にあるまとまりのあ

る選挙区を作成するためにも直接隣接していない行政区は少ない方が望ましい．この項は直接隣接していない行政区の発火数に応じて i 行 j 列のニューロンの発火を抑えるように働く．ただし，この項の値が大きくなりすぎるとニューロンが発火しなくなるので，定数 M で割ることで調整する．シミュレーションでは，区部は $M=2$，市部は $M=3$ に設定した．なお，シミュレーションではヒルクライミング項と同じ C を用いて強弱をつけた．

また，関数 f_3 は i 行のすべてのニューロンにおいて式 (4) が 0 であるかどうかを調べ，0 でなければ 1 を，0 ならば 0 を返す．式 (4) は制約条件を表した式であるが，これが 0 になるということは制約条件が満たされたということである．式 (10) において f_3 がない場合，i 行のすべてのニューロンが制約条件を満たしていても動作式が 0 にならないケースがある．したがって，そのまま計算していくとその状態を破壊してしてしまうこともありうる．それでは解への収束が保証されないので，i 行のすべてのニューロンが制約条件を満たした時点でこの項を 0 にする．この項は i 選挙区に所属する可能性の少ない行政区の発火を抑える働きもしているので，それまではこの項の値を反映させる．

3.6 シミュレーションの手法

今回のシミュレーションの手続きを以下に示す．

ステップ 1: まずはじめに繰返し数 t を 0 にセットする．繰返し数の最大値 $t\_\max$ を定める．$UTP=3$, $LTP=0$ にセットする．

ステップ 2: $U_{ij}(t)$ の値を $i=1\ldots n, j=1\ldots m$ までランダムな整数で初期化する．ここで n は選挙区数，m は行政区数である．

ステップ 3: $V_{ij}(t)$ の値を式 (2) を用いて $i=1\ldots n$, $j=1\ldots m$ まで求める．

ステップ 4: $\Delta U_{ij}(t)$ を求めるために式 (4), (7), (10) を計算する．ここで $\Delta U_{ij}(t)$ は，

$$\Delta U_{ij}(t)=\frac{dU_{ij}}{dt}. \qquad (12)$$

である．続いて，1 次オイラー法を用いて $U_{ij}(t+1)$ を求める．その式は，

$$U_{ij}(t+1)=U_{ij}(t)+\Delta U_{ij}(t). \qquad (13)$$

である．そして，$V_{ij}(t+1)$ を式 (2) を用いて求める．

この操作をすべてのニューロンの U_{ij}, V_{ij} $(i=1\ldots n, j=1\ldots m)$ について順に行う．

ステップ 5: もしすべてのニューロンにおいて $\Delta U_{ij}(t)=0$ であったならば，制約条件が満たされているということなので終了．あるいは $t=t\_\max$ ならば終了．そうでなければ t に 1 を加えてステップ 4 にもどる．

今回のシミュレーションでは，区部・市部どちらの場合も初期値は $-10\sim 0$ までの整数である．計算はすべて整数値で行われた．

3.7 シミュレーションの結果

シミュレーションはワークステーション DEC 3000 で行った．

今回は初期値を変えて区部・市部ともに 1000 回のシミュレーションを行い，繰返し数が 5000 回を越えたものに関しては収束しなかったものとした．**表 3** にその際の平均繰返し数，平均計算時間，収束率を記した．平均繰返し数は区部は 500 回程度，市部は 700 回程度であった．計算時間はプログラムの仕方にもよるので参考程度にご覧いただければよいが，繰返し数は市部の方が多いが，区部の方がニューロンの数が多いのでその分時間がかかっている．収束率は区部で 100%，市部で 99.9%であった．**図 4** に繰返し数のヒストグラムを示した．区部・市部双方のシミュレーションのほぼ 80%が 1000 回以内に収束していて，0~300 のところで特に頻度が高くなっているのが分かる．これらの値は初期値によって多少変わるが，他の初期値でも試してみたところ似たような結果がでているので，比較的少ない繰返し数で解に収束することが分かる．

図 5, **表 4** に結果例を示す．これは今回のシミュレーションで得られた区部・市部ともに一番格差の小さい結果である．これを見ると，最も人口が少ない選挙区は第 12 区で人口は 417488 人，最も人口が多い選挙区は第 1 区で人口は 534152 人である．この結果，東京都の人口格差は約 1.28 倍となった．実際の選挙区は約 1.47 倍であるので，約 0.2 格差を縮める結果を得られた．区部・市部別々に見ると，区部は実際の選挙区の 1.40 倍から 1.28 倍まで縮めたのに対し，市部では 1.47 倍から 1.14 倍にまで縮まった．実際の選挙区では人口が最大最少の選挙区は市部にあるので，市部で格差を大幅に縮めることができたことにより，全

表 3 シミュレーション結果：平均繰返し数・平均計算時間・収束率
Table 3 The result of the 1000 simulations: the average number of iteration steps and the average of the computation time (CPU time), and the convergence rate within 5000 iteration steps.

	平均繰返し数 (回)	平均計算時間 (秒)	収束率 (%)
区部	494	14.99	100
市部	713	12.15	99.9

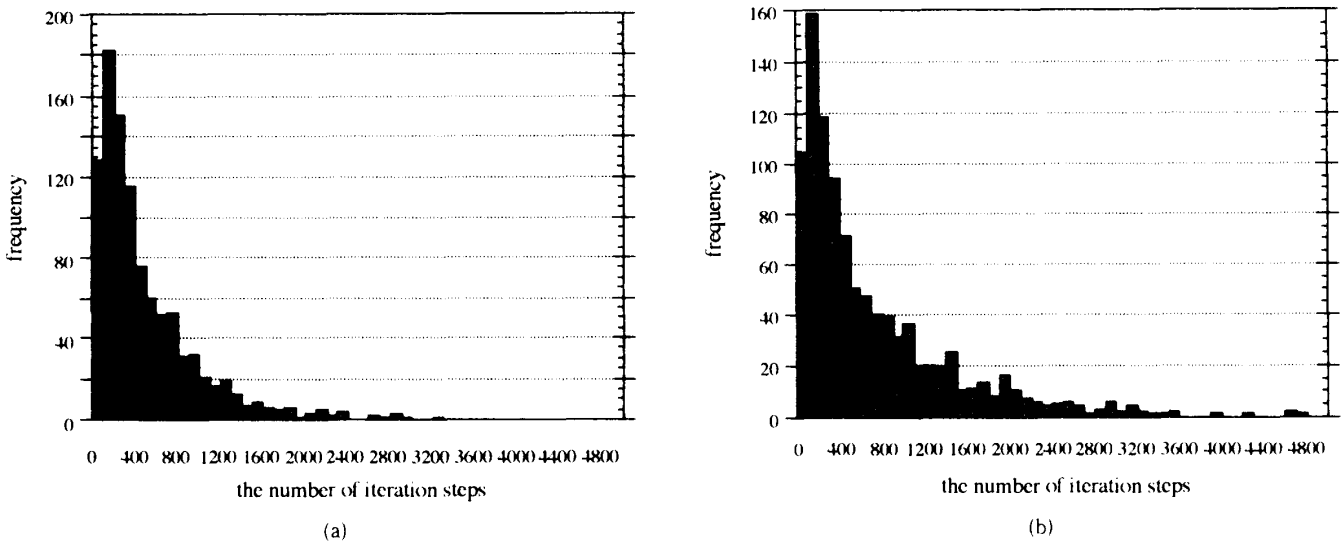

図4 繰返し数のヒストグラム．(a) 区部，(b) 市部
Fig. 4 Relationship between the frequency and the number of iteration steps to converge to solutions (a) for the urban part and (b) for the rural part.

表4 シミュレーション結果：図5のデータ
Table 4 The result of the simulation: the data of Fig. 5.

選挙区	行政区	人口（人）	選挙区	行政区	人口（人）
1	世田谷区 B	534152	18	立川市 昭島市 東村山市 東大和市	467330
2	中央区 墨田区 荒川区	475794	19	府中市 日野市 稲城市	433959
3	文京区 豊島区	443139	20	小金井市 小平市 国分寺市 国立市	436727
4	板橋区	518943	21	町田市 多摩市	493539
5	杉並区	529485	22	八王子市	466347
6	大田区 B	485325	23	武蔵野市 田無市 保谷市 清瀬市 東久留米市	490724
7	中野区 練馬区 A	463700	24	三鷹市 調布市 狛江市	437430
8	練馬区 B	474650	25	青梅市 福生市 武蔵村山市 秋川市 羽村市 瑞穂町 西多摩郡	433598
9	葛飾区 江戸川区 B	518107			
10	北区 足立区 A	497455			
11	千代田区 新宿区 台東区	499231			
12	大田区 A 世田谷区 A	417488			
13	足立区 B	488355			
14	江東区 島部	417495			
15	港区 品川区	503110			
16	江戸川区 B	472633			
17	目黒区 渋谷区	456847			

体で1.28倍という選挙区の区割りが可能となった．

シミュレーションでは，人口の上限と下限をそれぞれ540000人，400000人と設定しているので，これを満たしたものであればどんな組合せでも解としている．**表5**にシミュレーションで得られた人口格差とその頻度を記した．これを見ると市部では人口の制約条件が緩かったようで，かなりの組合せが見つかった（ただし，人口格差が同じでも組合せが同じとは限らない）．そのなかで格差が1.166249のときが623回で一番頻度が高い．区部では4種類の人口格差が得られ，格差が1.316025のときが721回で一番頻度が高い．今回のシミュレーションにおける設定では，これらの格差に

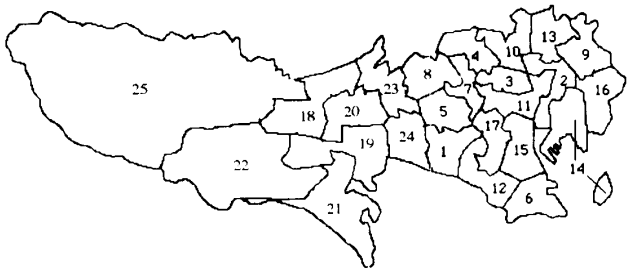

図5 シミュレーション結果
Fig. 5 One of the solutions (corresponding to Table 4).

おける組合せが最も収束しやすいものであるということになる．

表5 シミュレーション結果：人口格差と頻度
Table 5 The result of the 1000 simulations: Relationship between the ratio of the largest constituency to the smallest one and the frequency.

区部		市部	
人口格差	頻度	人口格差	頻度
1.279443	2	1.138241	1
1.309989	264	1.154423	1
1.313163	13	1.163667	6
1.316025	721	1.166249	623
		1.181025	14
		1.188489	78
		1.202256	71
		1.202273	1
		1.203547	4
		1.206729	1
		1.216200	1
		1.220708	1
		1.221461	10
		1.222426	1
		1.240843	7
		1.244754	27
		1.244871	3
		1.247475	14
		1.254558	2
		1.273604	1
		1.274457	47
		1.275702	45
		1.275794	38
		1.315090	1
		1.329927	1

今回は人口格差をできるだけ縮めると同時に，制約条件 (4) でなるべくコンパクトにまとまった選挙区を作成することも制約条件に加えた．ただし，今回のシミュレーションでは面積や距離などの地理的条件を考慮しているわけではなく，隣接関係のみでまとまりを考えている．そのため必ずしも地理的に見てよくまとまっているといえない場合もある．図2の実際の選挙区と図5のシミュレーションで得られた選挙区を比べてみると，実際の選挙区の第20区（市部）は4つ隣の行政区まで含まれている横長の選挙区になっているのに対し，シミュレーションでは，市部は3つ隣までとしたので，このような選挙区は含まれない．今回のシミュレーションでは実際の東京都の選挙区よりも近くの行政区どうしがまとまった選挙区を作成することができることが分かった．さらに地理的条件を考慮すれば，地理的にもまとまった選挙区を作成できるだろう．

最後に，動作式に付け加えた2つの項と C について述べる．ニューラルネットワークでは，初期値によっては極小解に陥ったり振動 (oscillation) を起こしていつまでも解に収束しないということが起こる．今回のシミュレーションではこれらの状態の発生をできるだけ抑えるために，動作式に式 (7), (10) の2項が付け加えられた．制約条件を反映した式 (4) のみでシミュレーションを行うと，収束率は数%であったのが，これらの項を加えることで表3に示したような高収束率を得ることができた．さらにこの2つの項に C で強弱をつけることで収束までの時間が短縮された．C に強弱をつけない（C はつねに1）で区部のシミュレーションを行ったところ，平均繰返し数1176回，平均収束時間35.62，収束率97.6%であった．表3の区部のシミュレーション結果と比べると，収束率はほとんど変わらないが，繰返し数が約2倍なので計算時間も約2倍かかった．強弱をつけたことにより2倍時間が短縮されたことが分かる．式 (4), (7) と C により安定した解への収束と高速化がもたらされた．

4. おわりに

本論文はニューラルコンピューティングによる小選挙区作成手法を提案した．これによって様々な条件のもとでの選挙区作成のコンピュータ化の可能性を示せたのではないかと思う．選挙区の作成にコンピュータを使う利点は，高速かつ正確に処理ができるというところにある．短い時間で解が出るので，初期値や制約条件を変えることで多種多様な区割りを行って比較検討することもできる．作業時間の短縮のみならず，区割りの質の向上も見込まれる．コンピュータですべてを決めろというわけではないが，コンピュータをうまく利用することで多くの国民の納得のいく質の高い区割りが行われることを，将来再び選挙区が変更される際には期待する．

今回のシミュレーションでは東京都の選挙区を対象にしたが，この手法を用いることで日本全国の選挙区の作成も行えるであろう．その際には，新聞にも取り上げられている問題でもある，各県に割り当てられている選挙区数の再分配も同時にできるのではないかと考えている．それによって2倍の格差の条件を満たした選挙区が作成される可能性もある．現在の選挙区では異なった県の行政区は同じ選挙区になることはないが，その枠を取り払ったときにどのような選挙区が作成できるか試してみるのも面白い．また，日本では一度選挙区が作成されるとその後再編成されるということはないが，アメリカでは選挙区がしばしば作り変えられる．アメリカの選挙区作成にこのアルゴリズムを応用して，それぞれの地域に特有の制約条件等を付加できる選挙区作成のアプリケーションを作ると役立つであろう．

参考文献

1) Hopfield, J.J. and Tank, D.W.: Neural Computation of Decisions in Optimization Problems, *Biological Cybernetics*, Vol.52, pp.141–152 (1985).
2) Takefuji, Y.: Neural Network Parallel Computing for Combinatorial Optimization Problems, *The Journal of Knoledge Engineering*, Vol.5, No.3 (Fall 1992).
3) Foo, Y.P., Takefuji, Y. and Szu, H.:Binary Neurons with Analog Communication Links for Solving Large-Scale Meeting (INNS) (1988).
4) Takefuji, Y.: *Neural Network Parallel Computing*, Kluwer Academic Publishers (1992).
5) 松本 元, 大津展之：ニューロコンピューティング, 培風館 (1992).
6) 武藤佳恭：ニューラルネットワークの組み合わせ最適化への応用, オペレーションズリサーチ, pp.324–330 (1992.7).
7) Takefuji, Y. and Lee, K.C.: Artificial Neural Networks for Four-coloring Problems and K-colorability Problems, *IEEE Trans. Circuits and Systems*, Vol.38, No.3, pp.326–333 (1991).
8) Takefuji, Y. and Lee, K.C.: An Hysteresis Binary Neuron: A Model Suppressing the Oscillatory Behaviors of Neural Dynamics, *Biological Cybernetics*, Vol.64, pp.353–356 (1991).
9) Hopfield, J.J.: Neurons, Dynamics and Computation, *Physics Today*, pp.40–46 (Feb. 1994).
10) Hopfield, J.J. and Tank, D.W.: Computing with Neural Circuits: A Model, *Science*, Vol.233, pp.625–633 (1986).

(平成7年5月9日受付)
(平成8年2月7日採録)

斎藤 孝之（学生会員）

昭和46年8月11日生. 平成7年慶應義塾大学環境情報学部環境情報学科卒業. 現在, 同大学大学院政策・メディア研究科修士課程在学中. ニューラルネットワークの組み合わせ最適化問題への応用に興味を持つ. 電子情報通信学会学生会員.

武藤 佳恭

昭和30年4月30日生. 1983年, 慶應義塾大学大学院博士課程電気工学修了. 工学博士（1983年）. 1988年からケースウエスタンリザーブ大学電気工学準教授. 1992年から慶應義塾大学環境情報学部助教授. Associate Editor of *Int. J. of Multimedia Tools and Applications*, MITI software CALS 幹事・セキュリティWG座長. 読売新聞コラムニスト, シンガポール・アルゼンチン・香港政府技術顧問. 現在ニューラルコンピューティングとCALS/ECの研究に従事. 著書・論文：「だれでもわかるディジタル回路」（オーム社, 1984年), 「CALS産業革命」（ジャストシステム, 1995年), *Neural Network Parallel Computing*（Kluwer Academic Pub., 1992), その他140編以上の科学論文（*Science*, Vol.245, pp.1221–1223, 1989）他, 受賞：情報処理学会20周年論文賞（1980年), NSF RIA 賞（1989年), IEEE Trans. on Neural Networks 功労賞（1992年), TEPCO Research 賞（1993～1995年), 神奈川アカデミー Research 賞（1993～1995年), 高柳 Research 賞（1995年).