特集　数理計画法

座談会

数理計画法をめぐって

L：予定していた参加人数に達したのでほっとしました．講師の方も含めて130名位になりました．

H：OR学会のなかではMP(数理計画)人口はわりと多いんですねえ．

K：今回は多くの人が興味をもつようなことをやさしく解説していただき，しかも，そのなかに最近のニュースまで含めていただきました．そのため，講師の方は多少やりにくかったかもしれませんが，参加者にとっては非常に有益だったのではないかと思います．

H：予稿集も大変ていねいに書かれており，これを読むとよく理解できて重宝しました．

線形計画法における誤差

L：MPのlarge scaleの問題を考えたとき，非常に精度を要求される制約式とそうでない制約式とがまじっている．つまり，ある等号制約は3～4けた，他の等号制約は1けた合えばよいということがよくあるのではないかと思います．そのような場合の対策といったものが考えられないでしょうか．

A：普通は誤差判定のパラメータを与えて，最適解が求まったあとにそれをoriginalの制約条件に代入して左辺と右辺の誤差がそのパラメータ以内ならばそこでやめるし，それ以上ならばreinversionします．そのパラメータ，誤差のtoleranceですが，それを各行ごとに変えるのはむずかしいかもしれません．

出席者：茨木俊秀(京大)・伊理正夫(東大)・岩村覚三(城西大)・岡本吉晴(三菱総研)・小野勝章(小野勝章事務所)・岸　尚(防衛大)・今野　浩(筑波大)・志水清孝(慶大)・鈴木正道(上智大)・高橋盤郎(早大)・富沢信明(東工大)・柳井　浩(慶大)・山下　浩(小野勝章事務所)・山本芳嗣(慶大)・渡辺　浩(筑波大)

記録：小島政和(東工大)

H：部分的には1けた合ってくれればよいというようなlarge scaleの問題をあつかう場合でも，シンプレックス法の演算のなかで行をmixするわけですから，全部の行が合ってくれないと困るというような態度でやらないと無理な気がします．つまり，部分的に精度を落とせる可能性がある問題でも，それを利用して計算の効率をupすることはできないでしょう．

K：結局，精度のよい行と悪い行がまざらないようなアルゴリズムにしなければいかんということだ．

H：しかし，まざらないようにする方法なんてないわけでしょう．

K：シンプレックス法ではだめだから他の方法を使わざるをえない．最初からエラーの話になってしまったが，性格が異なった2つのアルゴリズムで非常に大規模なLPを解いて答が一致するかどうかを確かめるというようなことをやっていますか．LPではよく知りませんが，system dynamicsではある計算機会社のDYNAMOと他のDYNAMOが異なった結果を出したという話を聞いたことがありますが．

G：4年位前に700×800位のLPをUMPIREとMPSXの両方で解いた経験があります．ほぼ同じ結果が出ました．optimal basisまで一緒で．

K：そうですか．UMPIREとMPSXはindependentにつくってあるんですか．

A：そうだと思います．けれども，プログラムを開発するときの最終段階でのtestで，敵との効率比較をするわけです．そのときに，同じ問題を自分の開発したプログラムと敵のやつで解いて比較しているはずです．その場合に，誤差等の関係で異なった答が出ることはあります．

F：私は500×4000程度の問題でテストしたことがあります．最適解は異なっていましたが，目的関数の値がほぼ一致したのでよいことにしました．詳細な解のようす

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

はだいぶ違っているようでした.

D：同じアルゴリズムでもコンピュータを変えると答が異なるということを経験しました．私の場合はその結果をIPに使いたかったので変数の値が問題になりました.

K：同じアルゴリズムというのはプログラムも同じということですか.

D：ええそうです．FORTRANで書かれたものです．word lengthが違っていたので，エラーに差が出て異なったbasisを選んでしまったのではないかと思います.

K：ごく単純な線形連立方程式でも，1000×1000位になると，ちょっとした丸め誤差の影響で解の形が変化する可能性は十分にあります．ＬＰの場合にはそれがbasisの選択に影響するから，見かけ上非常に異なった最適解を出すおそれがあります.

A：誤差はいくつかの時点でチェックしています. optimalやinfeasibleの決定のときにはかならずoriginalの制約に代入してチェックしているし，けた落ちは積和をとるときにいつでも調べています．その他にはπを使って求めたreduced costとtransformして求めたreduced costに有意な差が出た場合にはreinversionをやります.

大学では何を教えたら？

K：今日のシンポジウムの話をうかがっていて感じたことですが，そこまで技術が積み重なってくると大学で学生に何を教えたらよいか考えさせられます.

A：今日の話は非常に職人的な話なので，聴いている方にプログラムしていただかないかぎり，そのようなことをやっているということの紹介にしかなりませんでしたね.

K：そこまでやらないと大規模なものを解くときに役に立たないんでしょう.

A：少なくとも売り物にはなりませんね.

K：だから，教室で教えるのは線形計画法の原理だけで，“これだけでは実用にならんから，あとは現場で習いなさい”というほかはないですか.

H：おととしのTIMS京都大会のときに，ソフトウェア会社の人が近頃のLP codeの話をしたんですが，それに対して，大学でlarge scale LPを研究している人たちが“われわれは何をcontributeできるのか”という質問をしていましたね．その方面をやるようになったら，そのときに現場で訓練を受ければよいのであって，大学は基礎を教えるだけで，そのようなLP codeの話までも大学のなかで教えなくてもいいようです.

どの位の大きさのＬＰまで解けるの？

J：制約行列のnonzero要素のdensityの大きさによって解けるか否かは決まっているわけですか．たとえば1000×10000位の問題は解けることは解けるが，非常にdensityが低い場合だけなんですね.

A：そうです.

J：そうすると，“nonzero要素の個数が何個以下ならば解ける”といってもいいのですか.

A：いや，nonzeroといっても計算途中でのnonzero要素の個数ですから．それはoriginalの制約行列のnonzero要素にだいたい比例しますが，問題の構造にもよります.

D：与えられたＬＰ問題の変数の個数，条件式の個数とdensityから計算時間を出す経験式のようなものはないですか．条件式の個数の何倍であるとか.

A：計算センターでは見積を出さなければならないから“ある程度可能なんだ”と思います.

H：昔はiterationの数は条件式の数の2倍とかいっていましたね．userだと同じ構造の問題を繰り返し解くわけだから，異なったタイプのさまざまな問題を解く計算センターより見積りやすいでしょう.

整数計画法の費用の見積

F：ＩＰやＮＬＰ(非線形計画)に比較したらＬＰは費用が見積りやすいのではないですか.

H：integerは見積りができないから実用的でないということをいいますね.

D：あれはほんとうにできない．タイプが異なっていると費用がまったく違ってくるし…….

H：タイプが同じでも数字をちょっと変えると解けなくなるというようなことがよくありますね.

A：ですから，確定で見積らないで計算し終わった段階の費用を払うという契約にもっていくことが多いですね．とくに，ＭＩＰ(混合整数計画)の問題はちょっと見積れないので.

K：ＬＰでもＮＬＰでも掛けたお金に比例してよい答を出してくれる仕掛になっているんですかね．ＬＰは主単体法で解けば反復ごとに答が改良されるから，少なくとも単調増大ですな．integer programmingになると非常に不連続的に改良が行なわれるわけでしょ.

D：そうですね．branch and boundで解いたときには見つかるか否かで決まりますから，それに最適解が出た後もそれが最適であることをいうのに相当時間がかかる

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

のが普通です.

H：とにかく，掛けるお金には比例していない.

A：近似解法ならば少しは比例しているではないか．最近は，最適解を求めることよりもfeasibleな解のなかでなるべく目的関数の値の小さいのをというuserが多いんです.

整数計画はこれ以後も発展するの？

D：最初に頼まれたのは現状のsurveyだったのですが，現状でおもしろいというようなことがあまりないんですね．昔からあるアイデアがさらに細かくなったという感じです．アルゴリズムで目のさめるようなものはないですね．現在は反省期なのかもしれません．ちょっと一服といったところです.

G：一時期めざましく発展して，その後衰えて，また最近はそれでもなんとかなるという楽観的な意見が出ているということを聞きましたが，その辺はどうなんでしょうか.

D：その辺はよく知りませんが，そうなれば非常に嬉しいですね.

H：MPの大きいプログラムの中にMＩＰのアルゴリズムが組み込まれて，integer variableが20から30なら解けるという話がちらほら聞こえてきたりしますね．そんなことから楽観論が出ているのじゃないかな.

D：確かにMPSXは使いやすくなってきています.

L：ＩＰは実例を多く集めることが大切ではないかと思います．結局，変数の個数からは何もわかりません.

ＩＰを速くするには？

B：userがもってくる問題は0-1型が多いですか.

A：ええ，0-1型が圧倒的です.

B：違ったtypeの場合には0-1型になおすのですか.

A：いや.

H：0-1変数というのは“yes or no”なんだけれども，0-1型のIPをFORTRAN等でプログラムしたときには何ビットかを喰うわけです．そこで，プログラムのなかのinteger variableの部分だけをFORTRANから解除してもっとも原始的な姿の1ビットであつかってコンピュータのスピードの本領を発揮させたら，効率が3けた位upしてそれだけで一挙にぱっといくなんてことはないですか.

B：1ワード＝1ビットのマシンをつくったらどうです.

L：FORTRANなんかは非常に無駄をしているというか，ぜい肉が付きすぎているような感じがしますね.

K：どうですかねえ．損をしているとしても高々10倍位しか損をしていないでしょうね.

H：3けたまではいかんですか.

K：ええ，いくらがんばっても1けたですよ．それから，FORTRANというのは確かにぜい肉が付いているかもしれませんが，ある意味ではmachine independentなassembler languageという気がしますよね．あれはたいしたことやっていないですよ.

D：スピードが10倍位，100倍位になってもＩＰのなかで解ける問題の大きさはほんの少ししか増えない．それに関連するかと思うんですが，イリノイ大学にいったときにILLIAC 4というparallel computerでIPのbranch and boundのある種のプログラムをかけて評価したことがあるんですが，あまり速くなりませんでした．ＬＰですと非常に速くなるんです．ＩＰはsequentialにやるところが多いんですね.

A：ＬＰだと速くなるのはどうしてですか.

D：結局，行列計算なのでわけて計算できるからです．Gomoryの方法(cutting plane法)だと速くなるかもしれません.

G：branch and boundでもwidth firstに広げていけば速くなるように思えますが.

D：しかし，parallelといっても広げられる数がかぎられていますから.

K：シンポジウムでNP完全の説明の際に出てきたCHOICE命令をもつparallel computationの話は概念的なものなんでしょう.

D：そうです．実際にはつくれません.

ＩＰは解けるの？

K：solvabilityの話はおもしろかったですね．とくに，nonlinearなinteger programmingはそのformulationそのものが数学的にナンセンスだというのはおもしろいですね．よく，pseudo Booleanなんとかで解くという話がありますが.

D：あれは0-1にかぎっているからいいんですよね．ところが，原理的にはできても計算は大変です．どうしても最後は泥くさい話になります.

J：0-1型なら解けるというのならば，ＩＰの問題は2進表示すればすべて0-1型になるわけですから…….

L：有界ならばね.

D：今日のシンポジウムの話は有界でない場合も含んでいます.

J：だけど，有界でない場合は最初から考えてもしょう

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

がないんだから，一応有界として…….

H：世の中は有限だと思っている人と無限だと思っている人がいるから．

K：ＬＰとしては無限解をもつがＩＰとしては無限にはならんという場合があるんですな．整数点のすき間を制約条件がぬっているようなときは．

D：その場合にsolvableを証明するのはめんどうなんです．普通に多面体をつくったのではだめで，Presburgerの証明は場合わけして非常にうまく処理しています．しかし，あの作業はあのままでは無理で，実用的ではありません．ただ，理論的にできるというだけです．

ＮＰ完全性

K：数学基礎論のそういう立場から見て，こうすれば解けるという問題とＮＰ完全な問題とのgapは相当すごいんじゃないですか．

D：そうですね．そういう立場から考えるとＮＰ完全はずっと下のほうの階層にあるのですけれど，現実的にはそれ以上になるともうコンピュータではあつかえないという気がします．

H：それに比べると人間の頭はすごいんですねえ．travelling salesman 問題が絵に画いてあると最適解に近い道を画くでしょ．人間が直観的に道を画くときにはglobalに考えるからいいんですね．

G：phisicalに目で見たままが距離になっているような問題は割合に解きやすいんじゃないですか．

D：２次元平面に都市がばらまいてあって，Euclid 距離がそのまま距離になっているような問題は理論的にも比較的やさしいんです．ＮＰ完全にはなるのですけれど近似解に限定すればＮＰ完全でない問題です．ところが距離をもう少し一般的にしますと近似解を求めることすらＮＰ完全になります．そのような問題に対しては人間の直観がきかないということになるのかもしれません．

J：ＮＰ完全ならば多項式オーダーで解けないというのはほんとうなのですか．

D：証明はされていません．この分野で未解決な最大の問題です．

J：要するにＮＰ完全なやつを１つ多項式オーダーで解いてみせればいいわけですね．“そういうことをやってはいけない”といつもおこられているんですけれど．

K：多項式オーダーではだめなのかという問題設定にしますとね，もしかすると十分大きなkをとればn^kで解けるという可能性がないこともないわけでしょ．ただ，その場合kが１億であるとか，それこそ数学基礎論的な意味での多項式オーダーということになるかもしれませんね．おそらく，２乗とか３乗とかではだめなことは確かでしょ．

G：generalなＬＰが多項式オーダーで解けるか否かもわからないわけでしょ．

J：われわれが多項式オーダーという場合は２乗とか３乗とかであって，たとえば10乗とか100乗とかいう例はぜんぜんないわけですよね．われわれが多項式オーダーで解けると表現している問題は２乗または３乗までなわけですよ．だから，もうちょっとkをあげて10乗位だと相当いろいろな問題が解けるという可能性は十分にあるんじゃないですか．

K：そうでしょう．しかし，そういう例が１つもないというのはおもしろいですね．

J：それだけ幼稚なんだね．

D：nの何乗必要だという証明は非常にむずかしいんです．

J：いやそうじゃなくて，現実の問題でnの100乗位かかって解いたという例があればいいんです．そのような例はないわけですね．

D：近似解に関してはいろいろあります．

L：４乗以上でも何かありますか．

K：問題のサイズの測り方にもよります．「４乗か５乗位なら解けるけれど，それ以上改良しようと思ったがむずかしいのでやめておいた」なんてのはないことはないです．しかし，それも６乗位までででしょう．10とか20とかいうのはないですね．

G：制約式の数が５，６個だったら多項式オーダーで解けるというのがありますね．

D：そうですか？　何か制限が付いているのではないですか？　ナップザック問題ですらＮＰ完全ですから．

J：変数や制約の数が５，６個位の少ないときには，多項式オーダーといってもそのオーダーにしたがわないということになるんじゃないですか．

多目的最適化

H：多目的というなかには，いくつかの評価関数があるときにそれが最終的には１つにまとまるはずだという部分と本質的には１つにまとまらない面との２つが含まれていると思うんですが．つまり，一番上にトップの経営者がいてそれが最終的に判断を下すということになると，その経営者個人のなかでまとまる話ですね．逆に，ある問題を解いてそれを実行しようとして会社で提案したけれど，他の部から反対は出るしトップは調整してく

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

れないというような場面は，本来1つにまとまらないはずの多目的だという感じがします．

G：今日のはdecision makerが1人の場合ですね？ゲーム理論的なあつかいでなくて．

E：そうです．もっとも基本的な多目的システムというのはおたがいに独立に評価関数がある場合ですけれど，その場合にはパレート最適以上の最適性の概念はないでしょうね．

L：パレート最適というのは非常に有用だと思うんです．たとえば在庫管理をとりあげると，平均在庫量と平均リードタイムが問題になるわけですが，いろいろなシステムに対してそのなかのパレート解を出しておけばそれ以外のシステムははっきり悪いということがわかり，選択の範囲をせばめることができるわけです．

I：今日の多目的の話はOR的立場に立たなければいけない話だと思うんです．他の3つの話はOR的立場と数値解析的立場と二またをかけているようにうかがったのですが，ちょっとウィスキーの助けを借りていいますと多目的でもってどのように決定しようかということは決定者がいちばんいいたくないことだと思うんです．いったら手の内がすべてばれますから，OR的立場に立てば最後の最後までいわない種類のことだと思うんです．

H：いや，そうじゃなくて，概念形成ができてなくていおうにもいえないということがさきにあってね．たとえば，安いほうがいい，時間は短いほうがいいことはわかっていても，実際に値が示されないと実感が湧かない．そのような状況にある決定者に情報を与えて何かいわせるということを繰り返して，その本人により完全な情報を形成させるプロセス……．

K：今日の講演のGeoffrionのアルゴリズムはそのような状況を数学的に記述したものではないんですか．

E：そうです．潜在的な効用関数を前提にしています．でも，そのアルゴリズムによって求まる解がパレート最適であるとはかぎりません．パレート最適になるためには効用関数Φと目的関数fにかなりきびしい条件が必要です．

J：その本人の情報を引き出すといったって，普通はtransitivityが成り立っていないんですよね．

H：いや，僕のいっているのはトップが1人で総合化する人間がいる場合であって，それがいない場合の話とは違うんで，ただトップが情報不足で自分の評価関数を表現できない場合ですよ．まあ，transitivityが1人の頭のなかで成立していない場合もあるかもしれませんが．

J：いや，もちろん1人の場合でもtransitivityが成り立たないのが非常に多いということをいっているんですよ．

G：transitivityですけど，ある問題が与えられたとき，時間をかけて精密にチェックすると個人のなかでtransitivityが形成されていくという実験結果があります．

J：それはね，結局，“transitivityを満たさなければおれはだめだ”と思って努力するからそうなるんです．

I：多目的のいい点というのは“ある集合があってそのなかに入っていてそこからはみ出なければいいよ”というようなformulationにあるんじゃないかと思います．

K：だけど，多目的のむずかしいところは“このなかに入っていればいいや”という集合が集団あるいは個人によって異なり，場合によってはその共通部分が空集合になってしまうから，それをどう解決するかにあるわけですよね．つまり，多目的は相反する利害をもった個人がどうやって妥協するかというためのルールを見つけることになりますか．

H：やっぱり，主体が1つの場合と複数の話は別なんじゃないですか．

G：まったく別ですね．多目的計画は意思決定者が1人と仮定しているのとは違うのですか．

E：目的関数1つを意思決定者1人に対応させれば意思決定者は複数になるのですが，その場合でも彼らが協力し合って1つの解を捜すという問題になります．

G：conflictはない？

E：ええそうです．協力し合わないというのならゲームになります．多目的計画の一番むずかしいことは，スカラーの目的関数であれば2つの解のどちらがいいかを目的関数の大小で判断できるのですが，多目的ではそれができないという点にあります．

K：現実の問題に使おうというときに，“データをもとに計算すれば答が出るというように使うのか”あるいは“対人間的な交渉の過程で多目的計画の原理にのっとって合意を得るために使うのか”どっちなんでしょうか．

E：後者のほうに使いたいんですけどね．現状ではパレート最適解を出すだけで，その保証された集合のなかから合意を得る交渉は別に行なわれています．

ゲームは数理計画に入るの？

G：意思決定者が複数の場合にはゲームの理論と関係するわけですが，数理計画法に話をかぎればゲーム論的色彩のある問題は一応排除されると思うのですが，どうなんでしょうか．

H：僕は含めないできたんだけれど，多目的という言葉

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

を聞いてそのなかでおもしろいことというとどうしてもゲームにいくような気がするのですが.

L：含めたいですね. min か min max の違いだけですから.

H：いや，僕はゲームを数理計画に入れると数理計画にとって荷が重すぎるのではないかと思っているので，わけたほうがすっきりします.

E：僕はゲーム論も数理計画法と同一に考えています. ただ，特性関数を用いたゲームは形式が違うのでちゅうちょしますが，ゲームが制約式と複数の目的関数という形で表現されている場合には，目的関数を個々の意思決定者が独立にコントロールできることだけが多目的計画法との違いですね. 最適解であるための条件の表現もほとんど同じようになるわけです. ゲームでは単独な解が求まるような気がするかもしれませんが，一般的なN人ゲームでは均衡解は集合となってそのなかからどの解を選ぶかという状況は多目的計画の場合と同様です.

O：ゲームの core なんかもＬＰ の feasible region と同じで線形不等式の共通部分という感じですよね.

G：ゲームのごく一部でcoreを計算するフェイズは数理計画問題になりますが，ゲームにはゲーム特有のロジックみたいなものがありまして，たとえば経済学畑の人でゲームをやっている人はぜんぜん別の道具を使うわけです. その意味で数理計画と非常に離れた部分があるから，ゲームのすべてを数理計画に入れるのには抵抗を感じます.

E：それはおっしゃるとおりかもしれませんが，同じ問題でも数理計画法的に表現されていないということがあるかもしれませんね.

H：ただ，目的関数と制約条件という数学的な形になったそれ以後が数理計画だとは思わない. つまり，ＯＲとしての数理計画なんで，現実の問題を定式化するところから数理計画がはじまると思っています. 数理計画問題をつくるのも数理計画だと思っているから.

I：その議論は大変で，徹夜することになるからやめましょう.

非線形計画法の精度

I：今日のシンポジウムで非線形計画法の話をおもしろくうかがったのですが，あれも精度という点ではどうもうまくないなぁという気がします. 物理や統計の人が“精度が出ないで困る”といって問題をもち込んでくるのですが，どうも理由がはっきりしない. gradientを使ったりNewtonを使ったりして最適解らしいものは出る. けれども，どうも最後のパンチが効かない. 話をよく聞くと最適解の近所で目的関数がベターッとたいららしい. いわゆるなべ底なんです. そのためにどうしても精度が出ない. その辺はどんなものなんでしょうか.

C：それは代数方程式の重根のところで精度が出ないのと同じなんで，しょうがないんじゃないですか. もともと目的関数の値だけしか見ないとすればたいらなところならどこをとってもいいわけでしょう. それがよくないというのはそもそもformulationが悪いんじゃないですか. それはアルゴリズム以前の問題ですね.

F：問題のたちがいいというようにも思えますね.

I：そのとおりです. ＯＲの問題として目的関数をお金と考えれば“１円や２円違ってもおれが払ってやる”てなもんでどうってことはないんですが，数値解析的な完成ということを考えると，“目的関数が悪い”ではすまされない. 可能なかぎり手をうっておかないと具合が悪いという気がしてならない.

K：formulationが悪いということに関してですが，“なるべく生の原始的な姿に忠実に formulate しておく”，そして，“あまり高級な演算をほどこさずに答に到達するようなアルゴリズムを考えなければいかん”というこわれわれが現在得つつある経験則ではないかという気がとがするんですが. combinatorial な話，それからグラフ的な話なんかもあまり高級な概念をもちこむとたいていうまくいかない.

D：同感ですね.

H：その逆の場合はないですか. たとえばなべ底の部分を強調するような縦軸の変換をしたら近似精度が上がったなんていうことはないですか.

I：それはもちろん考えるべきですね.

非線形計画法の分類

H：nonlinear な問題のなかでどのようなものをあつかうかは人によって非常に異なる. 今日のシンポジウムの最後に，どのようなタイプならどの方法がいいかを整理して並べていただいた. あれは非常によかったと思います. けれども非線形計画の分類はあまりはっきりしていませんね.

I：とくに，数値的な性質に関する分類はなんにもない. nonlinear は linearに比べたらずっと広いわけだから，分類する必要がありますが，まだ分類の方法がよくわかっていない.

K：“目的関数を２次で近似することから脱却しなければいけない”ということをシンポジウムでおっしゃって

© 日本オペレーションズ・リサーチ学会. 無断複写・複製・転載を禁ず.

いたのですが，それは2を3，4のほうにしようということなのか，2を1にしようということなのか，どっちなのですか．両方ともむずかしいですよね．たとえば，絶対値なんかを使うと角ができて微分を用いたアルゴリズムが使えなくなるし，また3次4次はHessianが有効に働かないからむずかしい．

H：次数を1より小さくしてもむずかしくなりますね．

K：1より小さくなると，たとえば0.3次なんかになるとfixed charge programmingに近くなるわけですね．そのためにcombinatorialになる．

非線形計画法の応用

M：NLPの応用に関しては，“NLPのいいpackageがないからNLPに対するneedsがないんだ”という話と，“needsがないからいいpackageができないんだ”という話との両方ですくんでいる印象を受けましたが，その辺は？

B：それはNLPだけでなくOR全体にいえることじゃないですか．

K：positive feedbackがかかっているから，片方がしぼむと他方もだめになる．

H：今日はいろいろな問題を示していただいたわけですが，convexityなんかは？

B：めちゃめちゃですね．local minimumがたくさんあるので解くのをあきらめたという人もいるんですが，ほんとうにたくさんあるか否かもわからなくて．

K：convexとか2次とかうるさいことをいわれるのなら使いませんという人が多いんじゃないですか．うるさいことをいわずにやってくれなきゃだめですね．

B：それから，答をだしても材料の寸法と合わないから使わないという人がいるわけよ．土建屋さんに多い．

K：材料の寸法を制約に入れておいたら．

H：それはまさにinteger nonlinear programmingになる．

B：レンズ屋さんなんかの問題は大変ですね．たとえば，テレビの野球中継のための特殊な望遠レンズなんかは何十枚構成になります．ズーミングに使うんでかなり長い距離範囲で均一な像が要求されます．こんなのはNLPにとって絶好の問題です．local minimumがたくさんあるらしいといわれています．究極的な判断は職人的な勘に頼っている．もう1つレンズで問題なのは収差がいろいろあるので評価値が多くあることです．多目的になるわけですが，どのように解析しているかは知りません．

I：それこそレンズ会社のポリシーじゃないですか．非線形計画法のかなりのベテランがいて大きなprogrammingをあつかっています．けれどポリシーは極秘です．目的関数を教えたらだれでもできるわけですから．

H：日本で2台目か3台目に稼動したコンピュータは富士フィルムが……．

J：はじめてでしょ．

I：最近聞いた話ですが，目的関数を2次と認定される部分とそれ以外の部分にわけて，それぞれに異なったさまざまの手法を駆使しています．またテクニックの開発も盛んらしいです．

M：彼らはspecial purposeのNLPをやっているということですか．

I：ええ，でもspecial purposeだからといってばかにしていることはもちろんできなくて，むしろ目的関数の性質によって手法を使いわけていることには注目すべきです．

M：土木屋さんなんかはどうなんですか．

B：よくわかりませんが，問題意識としてはあるようです．

G：Interfaceにのっていたんですが，アメリカでは建設とか土木でNLPは割合に使われているようです．

I：僕が数年前にドイツにいったときに聞いたoptimizationに関する例題を2，3紹介しますとね．1つはミュンヘンのオリンピック・スタジアムに照明塔を付ける問題で，グランドの明るさが均一になるような照明方法．それから，slalom problemとよばれているものですが，大きなビル等があって客の多そうな所をぬっていくような地下鉄の設計．これはおそらく日本でもやっていると思うのですが，地図の問題．航空写真を何枚かとってそれを合わせて1枚の地図にする．アラビアの石油国が金を出してヨーロッパに頼んだものらしいです．もう1つは，テレビ塔の設計でコンピュータを使ってやっている．そんな話を聞きました．

K：essentialにnonlinearになるのは次元が異なっているものをあつかう場合に多いですね．たとえば，パイプの値段とそのなかを流れる水の量とか，そんなものは次元が違うから線形ではいかないですよね．

B：パイプの問題は1度きたことがあるんですが，やつらは予算をもっていないんでね．

M：お話もつきないと思いますが，そろそろ時間がきましたので終わりたいと思います．

今日はお忙しいところありがとうございました．（完）

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．