1 行列の扱い方ゼミ纏め

×：吐き出し法

しけたい４　(○：掃き出し法)

1　ブレイジングスター（r）– 前編

24 時間テレビで賽銭箱を隣に置かれることに巫女が拗ねたみたいだから、今回は私たちに出番はないかな・・ってあれ？題名は毎度どこかで聞いたことのあるような単語**に似たなにか**で踏襲されてたんじゃなかったのか？
今回はそのままの気がしてならないが・・・・

ふふふ。確かに今回のネタは解りにくいですね。多分、今まで作ったネタの中では最もわかりにくいネタでしょう。いずれわかりますよ・・・・
今回のネタは行列計算です。計算機シミュレートで最も基本的かつ重要なネタ。これが解けないと大抵のシミュレーションは出来ない。が、その一方でツールが整っていて自力で作る意味がしょっぱい科目でしょう。しょっぱいの一言でやる気をなくした方。頑張ってください。作る側も楽じゃないんです（何

誰かいたのかしら？
今日は貴女以外入ってこられないようにしたはずなのだけど・・・・

いや、多分しけたい４の幻覚だな。そうそう、話って何？

お金ある？

か、貸せるほどは持ってないぞ（逃げる準備をしながら）
というか、私も金欠だぜ。ツケがたんまり溜まってる。

大丈夫。貴女から奪う気は無いわ。実は私も眼鏡の店にちょっとツケがあってね。実際、ツケを払わなくていいくらいの貸しは、此処を守っているという意味ではあるのだけど、やっぱりこういうのって直接言い出しにくいじゃない？
そこで、ちょっと考えたのよ。

（中略）
なるほど。意図的にちょっとした事件を起こす。で、眼鏡に困ってもらって、その事件を解決する代わりにツケを無料にしてもらうわけだ。事件は意図的に起こしているわけだから賃金も何もない気はするが、春に雪が降ったりした事件の解決賃と考えればまあ問題ないって寸法か。いいね、私も手伝うよ。で、具体的には何をやるんだ？

狂言誘拐。
眼鏡の知り合いの貴女がさらわれたとすれば流石に黙っていないでしょ？
（縄を持ち出しながら）

待て、落ち着け・・・・
えっと、狂言誘拐はだな・・・・・
あ、私、冬休みの宿題があるから、時間の関係でだな・・・

（涼しい表情で）
貸して♪　直ぐに片付けてあげる♪

2　連立方程式を解くその１–掃き出し法

今までの講義で解ったと思うけど、差分法、有限要素法を用いた偏微分方程式の解法では、未知数、未知の固有ベクトルを求めるために必要な分の偏微分の差分近似や関数の内積を用意して行列式（または連立方程式）の形に直していたわね。
つまり、主に解きたい相手は**連立方程式**と**行列の固有値問題**ね。まずは連立方程式から考えていくわ。

掃き出し法は行列の形を問わずに使えるもっとも解りやすい解法の一つね。厳密な解を得られる**直説法**の一つだわ。直説法は基本的に計算に時間が掛かるから大きい行列が相手の時は、解に収束するようなアルゴリズムを繰りかえす**反復法**を用いるの。直説法で宿題になっているのは**掃き出し法**と **LU 分解**ね。どちらも N 行 N 列の問題を相手にしているわ。

掃き出し法一番解りやすいのは以下のような手順だぜ。わからない人は線形代数を補習だぜ！

- １列目について、a_{11} 以外が０になるように変換
- ２列目について、a_{22} 以外が０になるように変換
- 以下繰り返し

ただ、この場合 a_{nn} が０だったり限りなく小さい値だったときにうまく計算できないぜ。これを回避するために手順を以下のように書き換えるといい。

- a_{n1}（n=1,2・・）が最大となる行を１行目と入れ替える
- １列目について、a_{11} 以外が０になるように変換
- a_{n2}（n=2,3・・）が最大となる行を２行目と入れ替える
- ２列目について、a_{22}h 以外が０になるように変換
- 以下繰り返し

酷くシンプルな解説だな。というか、宿題早く終わらせていいのか、私は・・

3　連立方程式を解くその２–LU 分解

行列式を求めたり、連立方程式を解くのに最も実用的な方法の一つが **LU 分解**だ。言ってることは簡単なんだが、実際にプログラムを組むとちょっとだけ頭が痛くなるぜ。既存のアルゴリズムは余裕で存在するが、一度は作ってみるといいだろうな。

エッセンスはいたってシンプルよ。やるべきことは行列を以下の形に書き直すだけだわ。

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & \cdots & \cdots & a_{nn-1} & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ b_{21} & 1 & 0 & \cdots & 0 \\ b_{31} & b_{32} & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ b_{n1} & \cdots & \cdots & b_{nn-1} & 1 \end{bmatrix} \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdots & c_{1n} \\ 0 & c_{22} & c_{23} & \cdots & c_{2n} \\ 0 & 0 & c_{33} & \cdots & c_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 0 & c_{nn} \end{bmatrix}$$

右辺の行列を実際に計算してみると

$a_{11} = \mathbf{c_{11}}$

$a_{12} = \mathbf{c_{12}}$

・・・・

$a_{1n} = \mathbf{c_{1n}}$

$a_{21} = \mathbf{b_{21}}c_{11}$

$a_{22} = b_{21}c_{12} + \mathbf{c_{22}}$

・・・・

$a_{2n} = b_{21}c_{1n} + \mathbf{c_{2n}}$

$a_{31} = \mathbf{b_{31}}c_{11}$

$a_{32} = b_{31}c_{12} + \mathbf{b_{32}}c_{22}$

$a_{33} = b_{31}c_{13} + b_{32}c_{23} + \mathbf{c_{33}}$

・・・・

$a_{3n} = b_{31}c_{1n} + b_{32}c_{2n} + \mathbf{c_{3n}}$

とまあ、太字の文字のみが未知数なのだから、右辺の二つの行列の要素は上で長々と書いた計算を延々と繰り返せば求められることが解るわね。大サービス[*1]で計算方法を書いておくと

1 行目

c_{11} から c_{1n} までを a_{11} から a_{nn} にする

2 行目

$b_{21} = a_{21}/c_{11}$

ここから C 列に移行

$c_{22} = a_{22} - b_{21}c_{12}$

$c_{2m} = a_{2m} - b_{21}c_{1m}$

以下同様に

k 行目

b_{k1} から b_{kk-1} を求める

$b_{km} = \frac{1}{b_{mm}}(a_{km} - \sum_{p=1}^{m-1}(b_{kp}c_{pm}))$

c_{kk} から c_{kn} を求める

$c_{km} = a_{km} - \sum_{p=1}^{p=k-1}(b_{kp}c_{pm})$

以下 $A = LU$ と表記すると

$detA = detLdetU = c_{11}c_{22}・・c_{nn}$

と直ぐにわかるわね。連立方程式も L^{-1}, U^{-1} を求めれば解るわね。

[*1] ゼミで結構苦戦したなしなあ。

や、やけに今回は親切だな。
不気味でしょうがないんだが・・・

何言ってるの？
私はいつも新婚ほやほやのお嫁さんみたいに親切よ[*2]？
逆行列も分解と同様に逐一計算すると求められるわ。

$$\begin{bmatrix} 1 & 0 & 0 & \cdot\cdot & 0 \\ 0 & 1 & 0 & \cdot\cdot & 0 \\ 0 & 0 & 1 & \cdot\cdot & 0 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ 0 & \cdot\cdot & \cdot\cdot & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdot\cdot & 0 \\ b_{21} & 1 & 0 & \cdot\cdot & 0 \\ b_{31} & b_{32} & 1 & \cdot\cdot & 0 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ b_{n1} & \cdot\cdot & \cdot\cdot & b_{nn-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \cdot\cdot & 0 \\ d_{21} & 1 & 0 & \cdot\cdot & 0 \\ d_{31} & d_{32} & 1 & \cdot\cdot & 0 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ d_{n1} & \cdot\cdot & \cdot\cdot & d_{nn-1} & 1 \end{bmatrix}$$

まあ、見たとおりの式ね。未知数は d_{nm} よ。
行列の計算を逐一 ($k = 2, 3, 4$ と順番にやっていく) やれば

k 行目について

$\delta_{km} = \sum_{p=1}^{p=k} b_{kp} d_{pm}$
未知数は以下のアルゴリズムで求めればいいわ。($k > m$ のみ行えばいいわ。)
$d_{km} = -\sum_{p=m}^{p=k-1} b_{kp} d_{pm}$

全く同様に・・・

$$\begin{bmatrix} 1 & 0 & 0 & \cdot\cdot & 0 \\ 0 & 1 & 0 & \cdot\cdot & 0 \\ 0 & 0 & 1 & \cdot\cdot & 0 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ 0 & \cdot\cdot & \cdot\cdot & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdot\cdot & c_{1n} \\ 0 & c_{22} & c_{23} & \cdot\cdot & c_{2n} \\ 0 & 0 & c_{33} & \cdot\cdot & c_{3n} \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ 0 & \cdot\cdot & \cdot\cdot & 0 & c_{nn} \end{bmatrix} \begin{bmatrix} e_{11} & e_{12} & e_{13} & \cdot\cdot & e_{1n} \\ 0 & e_{22} & a_{23} & \cdot\cdot & e_{2n} \\ 0 & 0 & e_{33} & \cdot\cdot & e_{3n} \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ 0 & \cdot\cdot & \cdot\cdot & 0 & e_{nn} \end{bmatrix}$$

お察しの通り、今度は行列計算を逐一 n 行目からやっていくわ。

k 行目について

$\delta_{km} = \sum_{p=k}^{p=n} c_{kp} e_{pm}$
未知数は以下のアルゴリズムで求めればいいわ。($k <= m$ のみ行えばいいわ。)
$e_{km} = \frac{1}{c_{kk}} (\delta_{km} - \sum_{p=k+1}^{p=n} c_{kp} e_{pm})$

[*2] ハヤテの如く参照

逆行列を求められれば問題は解けたも同然だな。LU 分解や逆行列の計算で出てくる分母が 0 になってしまうと計算は失敗するぜ。分子が 0 の場合はもしかしたら値を 0 にしたら上手くいくかもしれない（実験はしてない）が、逆行列がない、つまり失敗した場合もあるので、フラグはしっかり立てておこう。

4 連立方程式を解くその３–ガウス・ザイテル法

先ほどちょろっと言ったけど、連立方程式の解を求める方法のもう一つに、反復計算を繰り返して近似解を得るというのもあるわ。こちらは初めから近似解を出そうと狙ったメソッドよ。
求めたい連立方程式の解 x を $Ax = b$
としたとき、以下のような試行を繰り返して行くの

行列の数列 x^i を作る
適当な初期ベクトルを定め、A を対角行列 U と上三角行列 L と下三角行列 D の和の形で書きなおし
$(L + D)x^{r+1} = b - Ux^r$
を満たすようにベクトルを更新していく

ちなみに、この場合
$x_i^{r+1} = (b_i - \sum_{j=1}^{i-1} l_{ij}x_j^{r+1} - \sum_{j=i+1}^{n} u_{ij}x_j^r)/d_{ii}$ となるわ。
この行列が収束するなら真の解に収束するのはわかるわね？具体的な式を見れば解るとは思うけど、この方法は対角成分が十分大きいときでないとうまく収束しないわ。

5 連立方程式を解くその４–共役勾配法

これも近似解をだす方法なのだが、こいつは結構難しいんだよな。ターゲットとなるのは対称行列で、尚且つ固有値が全て正のものだけだ。条件が限られていて「使えるのか？」と思う人も居るかもしれないが、今までの偏微分方程式の番組を見れば解るとおり、大抵の問題は固有値が正の対称行列っぽいことは予想がつくだろう。実際の工学の世界では最も使われる方法の一つだぜ。

この問題のポイントは、式の解となる x が
$|Ax - b|^2$
を最小にすることだ。上記の条件に従う行列 A については
$\frac{1}{2}x^T Ax - x^T b$ が min になるぜ。

証明はいたってシンプルだ。b を固有値 λ_k に対応する固有ベクトル $|k>$ で分解すると、真の解を x_0 として

$x_0 = \sum x_{0,k}|k>$

$b = \sum \lambda_k x_{0,k}|k>$

$\frac{1}{2}x^T Ax - x^T b = \sum \lambda_k (\frac{1}{2}x_k^2 - x_k x_{0,k})$

λ_k は正なので各々の k について最小を取るべきなので、x_k で偏微分すれば

$x_k = x_{0,k}$ で極小になることが解るぜ。

さて、具体的な計算に入ろう。先ほどと同じように反復メソッド

$x_{k+1} = x_k + a_k p_k$ しよう。

それと**残差ベクトル**

$r_k = b - Ax_k$

を定義するぜ。

x_0 は近似解や零ベクトルを取るといいだろう。また、$r_0 = p_0$ としておこう。さて、先ほど出した

$f(x) \equiv \frac{1}{2}x^T Ax - x^T b$ を最小にする a_k を考えよう。

$f(x_{k+1}) = f(x_k + a_k p_k) = \frac{1}{2}(x_k + a_k p_k)^T A(x_k + a_k p_k) - (x_k + a_k p_k)^T b$

$\quad = f(x_k) + \frac{1}{2}a_k^2 p_k^T A p_k - a_k p_k^T r_k$

a_k について偏微分すれば、極小となる a_k は・・・

$a_k = \frac{p_k^T r_k}{p_k^T A p_k}$

となる。よって、x_k, p_k が与えられれば x_{k+1} が求められるわけだな。ここまで来ると残る謎は p_2 から先をどうやって決めるかだな。とっととネタバレしてしまえば、p_k はこのようにつくる。

$p_{k+1} = r_{k+1} + b_k p_k$

$p_{k+1}^T A p_k = 0$

上の二つの条件を満たす b_k は

$(r_{k+1} + b_k p_k)^T A p_k = b_k (p_k^T A p_k) + r_{k+1}^T A p_k = 0$

$b_k = -\frac{p_k^T A p_k}{r_{k+1} A p_k}$

となるぜ。

さて、こう定義した理由なんだが、実は p_k, r_k はこう定義すると素敵な条件を満たしてくれるんだぜ。

$r_i^T r_j = 0 \quad (i \neq j)$

$p_i^T A p_j = 0 \quad (i \neq j)$

二つを同時に数学的帰納法で証明するぜ。

r_0 と r_1 が直行であることについて
$r_1 = r_0 - a_0 A p_0 = r_0 - \frac{r_0^T r_0}{r_0^T A r_0} A r_0$
$r_0^T r_1 = 0$
$p_0^T A p_1$ も同様に・・というか、定義よりこいつは 0 だ。

そして、ここから数学的帰納法の出番だ。$i, j <= k$ まで
$r_i^T r_j = 0 \quad (i \neq j)$
$p_i^T A p_j = 0 \quad (i \neq j)$
が成り立つとすると・・

$(i < k)$ について
$r_{k+1}^T r_i = (r_k - a_k A p_k)^T r_i = p_k^T A r_i$
$p_{i+1} = p_i + b_i r_i$ より、r_i は p_{i+1}, p_i の形で表せるので 0 になる。

(i=k) について
$r_{k+1}^T r_k = (r_k - a_k A p_k)^T r_k = r_k^T r_k - a_k p_k^T A r_k$
$p_k = r_k - b_{k-1} p_{k-1}$ より
$p_k^T A r_k = r_k^T A r_k$ なので
$r_{k+1}^T r_k = r_k^T r_k - a_k p_k^T A p_k = r_k^T r_k - p_k^T r_k$

$p_k = r_k + \sum_{i=0}^{i=k-1} c_i r_i$ で作られているので上の式は 0 になる。
これで $r_i^T r_j = 0 \quad (i \neq j)$ が示せたぜ。次は $p_i^T A p_j = 0 \quad (i \neq j)$ だな。
$(i < k)$ について
$p_{k+1}^T A p_i = (r_{k+1} + b_k p_k)^T A p_i = r_{k+1}^T A p_i = r_{k+1}^T (r_i - r_{i-1}) = 0$

$(i = k)$ については定義というか、p_{k+1} の作り方からして 0 だ。さて、これが求められてうれしいのは r_n が必ず 0 ベクトルになるってことだな。何せ r_0 から r_{n-1} までで互いに垂直なベクトルが n 個作られてしまっているからな。ということは、この方法は n 回繰り返せば厳密解になり、途中でやめても近似解になるというわけだ。収束のオーダーの近似とかはしないけど、結構素早く近似してくれるみたいだぜ。

5.1 この方法をやる前の下準備

上の二つの反復方法について、収束の精度を上げるため、以下のような技をくわえることも出来るわ。先ほどと同じように A を対角、上、下に分解したときの対角行列を用いて
$D^{-1/2} A D^{-1/2} y = D^{1/2} b$
$x = D^{-1/2} y$
見れば解るとは思うけど、全く同じ式よ。左辺の行列の対角成分が全て 1 になったのがポイントね。

これで収束が早まるの。それを考えるために
$Ax = b$ の b がちょっと動いたときの解の変化を考えてみようかしら。
b は A の固有値 d_i に対応するベクトル a_i の線形結合
$\sum c_i a_i$ と書き直せるわ。

すると、解 x は
$\sum (c_i/d_i)a_i$
になるわ。d_i がばらついていればばらついているほど、b の変化に対して解の変化が激しくなりそうなことが解るわね。
まあ、具体的な例を持ち出すと思いのほかそうとばかりはいかないのだけどね。

6　ブレイジングスター（r）– 後編

（縄を手に持って）
さて、宿題終わったわね。じゃ、早速始めようかしら？

な、なあ。こういうのって人質は優雅に茶でも飲みながらゆっくりしてちゃ駄目なのか？

やっぱり。ほら、こういうのってリアリティーがないと・・・・

がたっ、ばたばた・・・・
どすん・・・・
（数分後）

（以下バリトン声で）
あー、もしもし。有隣堂ですか？

いいえ。有隣堂ではありません。

お前の店の常連客の魔法使い君を誘拐させてもらった。返してほしくば身代金を払っていただこうか。おっと、困ったからといって巫女とか妖怪とかの類に相談するのはやめてもらおうか。その場合、人質の無事は保障しない。

そ、それは大変だ。出せる限りのお金は出そう。身代金はどうやって支払えばいい。それと、魔法使いは本当に無事なんだろうな。声を聞かせてくれ。

たーすーけーてー（じたばた）
ちなみに、眼鏡の顔グラ提供＝ **LostRage** 様です。

ど、どうやら無事そうだな。

身代金は朝 8 時に神社の賽銭箱に入れてもらおう。賽銭箱の近くには人質の居場所を書いた地図をがあるから、身代金を払い次第、お前は人質を回収しに行け。現場にこそこそ残ったり、身代金を払わなかった場合人質の無事な姿が見られると思うなよ？

言いたいことはそれで全部か？

ああ。何か確認したいことはあるか？

お前は、重大なミスをひとつ犯した！

何っ！？

神社には極めて賽銭にがめつい巫女がいるのだよ。どう頑張っても賽銭箱の近くでお金の音がしたら瞬く間に回収されてしまうね。

だ、大丈夫だ。その巫女とやらが気づく前に回収すればいい。なんなら時間をずらしてもいいのだぞ・・・

無駄だな。その巫女の賽銭への執念は凶悪犯のお前でさえも想像がつかないレベル。お前を遥かに凌ぐであろう凶悪さだ。悪いことは言わない、別の場所にするんだ。

・・・・・・・・・・

ぶっ・・・・

（棚から「何か」を取り出しながら）
どうやら自分の立場をわかっていないようだな。これ以上無駄な抵抗をするようなら人質がどういう目にあうか解っていないというのなら教えてやろう。

ちょ、待て。それは洒落にならな・・・
ぎゃあああああああああああああああああああああああああああああ

あああっ、くそっ、こういうことか。
Braze=蝋付けをする
じゃないか。あの題名、英語で綴ると BrazingStar って読むわけだ。

またしばらくしたら連絡する。

って、あれ？
なにゆえ、そんな良くわからない縛られ方してるの？

お前がやったからだろー。
縛り方の固有ベクトルがわからんとどうにも脱出できそうもないぜ。ひとまずこれ、解いてくれぇ。

えー。面倒臭い。
・・・わ、解ったわよ。今回のお相手は N 行 N 列行列の固有値問題ね。この手のメソッドで有名どころは**ハウスホルダー法**、**QR 法**、**二分法**、**ランチョス法**、**緩和法**あたりね。ここでは QR 法以外を扱うわ。

7 ハウスホルダー法による対角行列の変形

ハウスホルダー法はあるベクトル r を回転させて、ある軸にそろえさせたいときに使えるんだ。r の大きさと同じ大きさで、かつ使いたい軸にそったベクトル p を用意しよう。この時 r を p にするような行列を探せば良いと解るな？

そんな行列は以下のように定義できるぜ。$H = I - \frac{2(p-r)^T(p-r)}{|p-r|^2}$ 図にすると解りやすいかな。H は対称行列で 2 乗は I になるぜ。これによって、この変換でも固有値が変わらないことが解るはずだぜ。

ハウスホルダー法用いて、ひたすら**対角行列** A を三重対角行列に書き換えるぜ。行列変換に使う時は対角のみだから注意してくれ。例えば 4 行 4 列ならまず 1 行目のベクトル $(a_{11}, a_{12}, a_{13}, a_{14})^T$ を変形したい。ベクトル $(0, a_{12}, a_{13}, a_{14})$ の大きさを a とすれば最終的には $(a_{11}, a, 0, 0)$ の形になれば良いわけだ。早速、先ほど紹介した方法を用いて変換するぜ。具体的には HAH を求めれば OK だ。

A,H が共に対称行列だから、$a_{13}, a_{14}, a_{31}, a_{41}$ の部分が 0 になった対称行列が完成するぜ。この変換はただの回転だから固有値に変化は生じないぜ。あとは繰り返しだ。具体的にはこんな感じで変換されていくぜ。

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{12} & a_{22} & a_{23} & a_{24} \\ a_{13} & a_{23} & a_{33} & a_{34} \\ a_{14} & a_{24}a_{34} & a_{44} & \end{bmatrix}, HA = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a & b_{22} & b_{23} & b_{24} \\ 0 & b_{32} & b_{33} & b_{34} \\ 0 & b_{42} & b_{43} & b_{44} \end{bmatrix}, HAH = \begin{bmatrix} a_{11} & a & 0 & 0 \\ a & c_{22} & c_{23} & c_{24} \\ 0 & c_{23} & c_{33} & c_{34} \\ 0 & c_{24} & c_{34} & c_{44} \end{bmatrix}$$

（巫女がうつらうつらしてるのに気付いて）
こら、寝るなぁあ！

8　固有値を求める–2 分法

さて、こうして得られた三重対角行列（A とする）から固有値を求めるわ。
$det|A - \lambda I|$
は三重対角行列の性質から数列を作って簡単に求めることが可能よ。

$\phi_0 = 1$
$\phi_1 = a_{11} - \lambda$
$\phi_{k+1} = (a_{k+1,k+1} - \lambda)\phi_k - 2(a_{k+1,k})^2\phi_{k-1}$

ϕ_i はここで面白い性質を満たしているのよ。それは
ϕ_i を 0 から順番に n まで見て行ったとき、符号の反転する回数は、λ より行列 A の固有値の数に等しい
これが本当なら、以下のような手順で固有値を求められるわね（スツルムの定理の応用。簡単な証明は下に）

- 最も大きい固有値を含んでいる領域 [a,b] を定める
- $det|A - (a + b/2)I|$ を求める。
- ϕ_i の符号反転回数が 1 以上なら領域を [(a+b)/2,b] にする
- ϕ_i の符号反転回数が 0 なら領域を [a,(a+b)/2] にする
- 繰り返して領域を絞る

スツルムの定理の簡単な証明：
まず ϕ_i について考えるわ。数列自体が ϕ_{k+1} をユークリッドの互除法で ϕ_k, ϕ_{k-1} としていると解釈できるわね。もし、三重対角行列に 0 がなければ、任意の ϕ_i は他と同時に 0 になることは無いわね。
$0 < k < n$ について
$\phi_k(\lambda)$ が ϵ、0、$-\epsilon$ となるような微小な領域について λ を動かすとするわ。ϵ は任意の正の実数ね。

さて、ϵ は何処までも小さくできるので λ の値が動く間に『ϕ_i 以外の符号が入れ替わらない』という状況を作れるわね。そして、ϕ_i の式を見直すと

$\phi_{i+1} = (a_{i+1,i+1} - \lambda)\phi_i - 2(a_{i+1,i})^2\phi_{i-1}$

つまり、$\phi_i = 0$ の時は ϕ_{i+1}, ϕ_{i-1} は（正、負）か（負、正）。そして λ の移動中はこれらの符号は入れ替わらないとしているから（下の表での ϵ_0 は任意の絶対値が小さい数）

x	ϕ_{i-1}	ϕ_i	ϕ_{i+1}	符号変化数
$\lambda_0 + \epsilon_0$	$\pm$	+	$\mp$	1
λ_0	$\pm$	0	$\mp$	1
$\lambda_0 - \epsilon_0$	$\pm$	-	$\mp$	1

こうなるわね。つまり、符号の入れ替わった回数は変わらないわ。一方 k=n の時は

$\phi_n = (a_{n,n} - \lambda)\phi_{n-1} - 2(a_{n,n-1})^2\phi_{n-2}$

$\phi_n = 0$ が正から負へ変化しても、ϕ_{n-1}, ϕ_{n-2} などの符号は変化しないのだから、符合の変化回数が変わるわ。さて、ϕ_i は λ の i 次の関数で、i 次の係数は $(-1)^i$ だわ。これから $\lambda \to -\infty$ とした時、ϕ_i は全て正になるはずね。

同様に $\lambda \to \infty$ とした時は正、負、正、負・・となるはずだわ。以上から、ϕ_i **を 0 から順番に n まで見て行ったとき、符号の反転する回数は、λ より小さい行列 A の固有値の数に等しい**という命題が成り立つと証明できたわ。重解の場合などはちょっと考えるのが面倒だけどこれを応用すれば解けるはずよ。細かい証明は・・・省略で。

8.1 コラム–固有値から固有ベクトルを求める

ランチョス法、緩和法については固有ベクトルも同時に求められるからいいのだけど、二分法については固有値から固有ベクトルを自力で出さなければいけないのよね。いくつか方法があるらしいけど、それなりに簡単なのが、欲しい固有値 λ に対応する固有ベクトルと直行でない適当なベクトルに

$\frac{1}{A-(\lambda+\epsilon)I} = (A - (\lambda + \epsilon)I)^{-1}$　(I は単位行列) をかけることね。

$(A - (\lambda + \epsilon)I)^{-1}x = y$ として、x, y を固有ベクトルで分解すると

$x_\lambda|\lambda> + \sum_{i \neq \lambda} x_i|i> = \sum_i (i - (\lambda + \epsilon))y_i|i>$

上の式の i は固有値に対応する添え字。また、上の式では両辺に $A - (\lambda + \epsilon)I$ をかけたわ。

y_λ は $\epsilon \to 0$ で ∞ に飛ぶことから、固有値に対応した固有ベクトルが求められそうね。

もっとも、ϵ が 0 に近づけば近づくほど逆行列出すのが厳しくなるのだけどねー。$\epsilon = 0$ だと逆行列なくなっちゃうし。固有ベクトルまで欲しいのだったらやっぱりランチョス法とかがオススメね。あっちの方が性能よさげだし（ぇ

9　ランチョス法

ものすごく巨大な N 行 N 列行列 A と戦う時に物凄く強力なメソッドがこのランチョス法よ。ハウスホルダー変換などと違い、固有値をいくつ求めたいのか、どのくらいの精度で求めたいのかによって計算回数を変えられることが最大のポイントよ。

簡単に手順を説明すると以下のとおりになるわ。

1．規格化された初期ベクトル u_1 を作る

2．$Au_1 = \alpha_1 u_1 + \beta_1 u_2 \quad u_1^T u_2 = 0 \quad |u_2| = 1$ を解く

3．$Au_2 = \beta_1 u_1 \alpha_2 u_2 + \beta_2 u_3$（$u_3$ は u_1, u_2 と直行で規格化されている）を解く

4．$Au_k = \beta_{k-1} u_{k-1} \alpha_k u_k + \beta_{k+1} u_{k+1}$（$u_{k+1}$ は u_1・・u_k と直行）を解く

5．ある程度これを繰り返し（m 回）て、飽きたら終了する。

6．u_1・・・・u_m が互いに直行していれば、行列計算は**一意に**[*3]以下のようになるわ。

$$Au_1 = \begin{bmatrix} u_1 & u_2 & u_{13} & \cdot\cdot & u_m \end{bmatrix} \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \cdot\cdot & 0 \\ \beta_1 & \alpha_2 & \beta_2 & \cdot\cdot & 0 \\ 0 & \beta_2 & \alpha_3 & \cdot\cdot & 0 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ 0 & \cdot\cdot & \cdot\cdot & \beta_{m-1} & \alpha_m \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ \cdot \\ 0 \end{bmatrix}$$

上の式は m が小さい時は厳密には成り立たないの解るかしら。理由は Au_m を考えればいいわ。上の式の右辺には $\beta_{m+1} u_{m+1}$ の部分がないのだから明らかね。まあ、n 回繰り返せば u_{n+1} は 0 ベクトル以外を取れないのだから厳密に成り立つのだけどね。

さて、三重対角行列が出来たあたりで気付いた人も居ると思うけど、この m 行 m 列の行列の固有値を求めるとあら不思議、N 行 N 列の行列の固有値のうち**絶対値の大きい m 個の近似値が得られる**の。以下のように考えると納得がいくわね。いちいち行列を書くのは面倒だから、上の式を

$Au_k \fallingdotseq \vec{u}^T B e_k$

（e_k は k 成分だけ 1 となるベクトル。$\vec{u}$ は右辺のベクトルたち。B は行列）

と書き直そうかしら。

$A^n u_k \fallingdotseq \vec{u}^T B^n e_k$

u_k が絶対値が A の最大の固有値 c_{max} に対応する固有ベクトル x を含んでいれば、n を大きくすると左辺は

$c_{max}^n c$ のスカラー倍に収束するわ。

同様に右辺について考えると右辺は B の最大固有値 b_{max} と固有ベクトル b を用いて

$b_{max}^n \vec{u}^T b$ のスカラー倍に収束するわ。

両辺が等しいという要請から

$b_{max} \fallingdotseq c_{max}$ が成り立つわね。

[*3] 直行していないと、右辺の行列部分が一意に定まらない。

求める固有値の精度は、m が大きいほどよくなるわ。さて、次は α, β の求め方を考えて、実際にランチョス法のアルゴリズムを考えるわよ。とかいいながらも、式が求められた時点で天下りもよい所なのだけどね・・・・

突然面倒くさくなったからカンペ公式を貼っておくわ。はい、こんな感じ～

$\alpha_1 = u_1^T A u_1$

$\beta_1 = \sqrt{(Au_1 - \alpha_1 u_1)^T (Au_1 - \alpha_1 u_1)}$

$u_2 = \frac{1}{\beta_1}(Au_1 - \alpha_1 u_1)$

$\alpha_k = u_k^T A u_k$

$\beta_k = \sqrt{(Au_1 - \beta_{k-1}u_{k-1} - \alpha_1 u_1)^T (Au_1 - \beta_{k-1}u_{k-1} - \alpha_1 u_1)}$

$u_{k+1} = \frac{1}{\beta_k}(Au_1 - \beta_{k-1}u_{k-1} - \alpha_1 u_1)$

式の形から

$Au_k = \beta_{k-1}u_{k-1} + \alpha_k u_k + \beta_k u_{k+1}$ が成り立つことは明らかね

u_k の直行性についても、u_1 と u_2 が直行であることを計算（流石に出来るわよね）から数学的帰納法を用いればいいわ。$i, j <= m$ について $u_i^T u_j = \delta_{ij}$ と仮定して、$u_{k+1}^T u_i$ を計算すると

$i < m-1$ の場合

$u_i^T u_{m-1} = 0 \quad u_i^T u_m = 0 \quad u_i^T A u_m = u_m^T A u_i = 0$

$i = m-1$ の場合

$-\beta_{m-1} u_i^T u_{m-1} = -\beta_{m-1} \quad u_i^T u_m = 0 \quad u_i^T A u_m = u_m^T A u_i = \beta_{m-1}$

$i = m$ の場合

$\beta_{m-1} u_i^T u_{m-1} = 0 \quad -\alpha_i u_i^T u_m = -\alpha_i \quad u_i^T A u_m = \alpha_i$

以上より u_{m+1} は u_1 から u_m と直行するわね。

見ての通り、逐一計算するだけでどんどん m を大きくしていくことが出来るわね。これがランチョス法の強みなの。ランチョス法を用いた固有値計算は、

１．m をある程度まで大きくする

２．三重対角行列の固有値を求める

３．m に対して、求めたい固有値が収束するまで繰り返す

この手順で固有値を求めていくわ。疎行列が相手の場合でもとても使い勝手のいいメソッドよ。

計算時の注意点は二つあるわ。一つは u_m の直行性をしっかり保っておくこと。新しいベクトルを出す毎に、丸め誤差などで入ってくるほかのベクトルの成分はしっかり追い出しておく必要があるわ。もう一つは u_1 が A の固有ベクトルと直行していた場合ね、そのベクトル成分が任意の u_m を含まなくなってしまうからちょっと注意する必要があるわ。

さて、これで縄はほどけたはずだけど・・・・
精度が足りないわね。

げ、繰り返し計算で積み重なった誤差で解けないってどういう縛り方だよ・・・
じゃあ緩和法を使うしかないかぁ・・

10　緩和法

緩和法のエッセンスはびっくりするぐらい単純だぜ。この方法では行列 A に対して固有値最小の固有ベクトルを求めたいわけなんだが、固有値が最小の固有ベクトル x は
$\rho = \frac{x^T A x}{x^T x}$（レイリー商と呼ぶ）
を最小にしてくれるってのは解るな？
任意のベクトルを固有ベクトルで分解すれば直ぐに解けるぜ。

緩和法の手順はいたってシンプル。以下のとおりだぜ。

1．直行化基底ベクトル u_1・・・u_n を用意する
2．初期ベクトル $x_0 = \sum c_i u_i$ を作る
3．レイリー商を c_1 の関数として極小値となる c_1^* を求める
4．$x = c_1^* u_1 + \sum_{i=2}^{i=N} c_i u_i$ とする
5．c_2 以降にも同様の処理を施す
6．c_n まで繰り返すと固有ベクトルが出る・わけではないので更に繰り返す。
7．ベクトルがある程度収束したらやめる。

$x_0 = \sum c_i u_i \quad x(\alpha) = x_0 + \alpha u_1$ とした時
$\rho(\alpha) = \frac{x_0^T A x_0 + 2\alpha(x_0^T A u_1) + \alpha^2 u_1^T A u_1}{x_0^T x_0 + 2\alpha c_0 + \alpha^2}$
$\frac{d}{d\alpha}\rho(\alpha) = \frac{1}{(x_0^T x_0 + 2\alpha c_0 + \alpha^2)^2}(x_0^T x_0)(a\alpha^2 + b\alpha + c)$
$a = \frac{u_1^T A u_1 c_1 - f}{x_0^T x_0} \quad b = u_1^T A u_1 - \rho(0) \quad c = f - \rho(0)c_1 \quad f = x_0^T A u_1$

ここから、二次関数の解を求めればよいことが解るぜ。さて、二次関数の解を求めるのはたやすいが、極小値ってどう考えればいいんだ？
この問いに答えるために α を $+\infty, -\infty$ に飛ばしたときのことを考えよう。すると、どちらも同じ値に収束していくことがわかるだろう。
このことから、a が正の時は大きいほうが、a が負の時は小さいほうが解になると解るぜ。

さて、二次関数の解はおなじみの
$\frac{-b\pm\sqrt{b^2-4ac}}{2a}$ で表せるが、実は
$\frac{2c}{-b\mp\sqrt{b^2-4ac}}$ とも書くことが出来る。
なんでこんなのを使うのかと言うと、先ほど求めた c について考えると、x_0 が求めたい固有ベクトル（固有値 λ）に十分近い場合
$c \to \lambda x_0^T u_1 - \lambda u_1 = 0$ と 0 に収束するからなんだ。

c が 0 に近づいてくると、**桁落ち**の問題が無視できなくなってくる。おなじみの表記方法だけだと、$-b\pm\sqrt{b^2-4ac}$ が 0 に近づいてくる時に不味いわけだ。$-b\pm\sqrt{b^2-4ac}$ が 0 に近づいてくるときは $-b\mp\sqrt{b^2-4ac}$ は 0 に近づかないからそっちを使えってわけだな。b が正か負かによってしっかり分岐することを忘れないで欲しいぜ。

こうして、無事に α を設定できたら x と ρ を更新するわけだ。x の更新はラクチンだから良いとして、ρ の更新にくれぐれも、$\rho(\alpha)$ の再計算はやらないでくれよな。計算すればわかることだが
$\rho(\alpha) = \rho(0) + \frac{(b\alpha+2c)\alpha}{x(\alpha)^T x(\alpha)}$
で求められるからな。こっちの方が大分計算が楽だぜ。

最後に励起状態の求め方を説明しておくぜ。緩和法によって目出度く固有値の絶対値が最も大きい固有ベクトル $|k>$ が求められたら
$A' = A + S_k|k><k|$ とすればいい。
A の任意の固有ベクトル $|i>$ に対して
$A'|i>=(\lambda_i+\delta_{ik}S_k)|i>$ となるわけだから、S_k を適当な値にすれば A' 最も固有値の絶対値の大きい固有ベクトルは $|k>$ 以外の A の固有ベクトルになってくれるわけだな。

このメソッドも延々と繰り返すことが可能だぜ。いうまでもないといえばいうまでも無いけどな。固有ベクトル分解すれば直ぐに納得がいくと思うぜ。緩和法の弱点は固有値が近い固有ベクトルがあった場合、収束の精度が悪くなることだな。レイリー商を用いてる時点でしょうがないことなのだけどな。
っと、やっと縄が解けたぜ。

11 QR 法

さて、無事に縄もほどけたことだし、計画を再開しようかしら。賽銭箱に身代金が入ったことを確認し次第、人質のありかを書いた地図を送る。地図は QR 法を使えば読めるような行列の固有値問題にすればいいわね。

えー、今まで出した方法は使わないの？

教育的じゃないから駄目よ。
さて、QR 法のやり方を眼鏡に説明するための手紙を作るわよ。新聞切るの手伝って。

あいよっと。QR 法ってのは、これまた対称行列の固有値を求めるのに使うメソッドだ。このメソッドの最大のエッセンスは、三重対角化させた対称行列を直行行列（ほとんど回転行列ばかりだが）の積 Q と上三角行列 R の積に行列を分解することだ。方法はシンプルだが、証明が面倒なんだよな・・

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{12} & a_{22} & a_{23} & 0 \\ 0 & a_{23} & a_{33} & a_{34} \\ 0 & 0 & a_{34} & a_{44} \end{bmatrix} = Q \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ a & c_{22} & c_{23} & c_{24} \\ 0 & 0 & c_{33} & c_{34} \\ 0 & 0 & 0 & c_{44} \end{bmatrix} \equiv QR$$

Q は直行行列の積だから $Q^T = Q^{-1}$ よ。殆ど回転行列ばかり使うみたいだから、回転行列で想像しちゃってもいいわ。ここはしっかり納得してね。
さて、あっさりネタバレしちゃうと QR 分解はこのような手順でやっていくわ。実際は直行行列ならいいらしいけど、回転行列が一番楽よ。

1．三重対角行列に回転行列を掛けまくり、$A = QR$ を作る
2．$A_1 = RQ$ を求める。
3．$A_1 = Q_1 R_1$ を求める
4．$A_2 = R_1 Q_1$ を求める
5．これを繰り返すと、A_k は対称行列へ向かっていく。
消したいのは上の例では a_{21}, a_{32}, a_{43} の三つなので、順番に一つ一つ消していくわ。大体こんな感じ。

$$P_1 = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad P_1 A = \begin{bmatrix} a'_{11} & a'_{12} & a'_{13} & 0 \\ 0 & a'_{22} & a'_{23} & 0 \\ 0 & a_{23} & a_{33} & a_{34} \\ 0 & 0 & a_{34} & a_{44} \end{bmatrix}$$

$$P_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad P_2 P_1 A = \begin{bmatrix} a'_{11} & a'_{12} & a'_{13} & 0 \\ 0 & a''_{22} & a''_{23} & a''_{24} \\ 0 & 0 & a''_{33} & a''_{34} \\ 0 & 0 & a_{34} & a_{44} \end{bmatrix}$$

なんで。どうしてよ。どうして収束するのよと思った貴女。正しいです。しけ４も同じことで悩みました。ここからはそれの証明を延々とやっていきますので御安心を。

$A = QR$
$A_1 = RQ = Q^T AQ$
$A_k = Q_k R_k$
$A_{k+1} = Q_k^T A_k Q_k$
より、A_k の固有値が A と同じであることと
$A_k = Q_{k-1}^T Q_{k-2}^T \cdot\cdot QAQQ_2 \cdot\cdot Q_{k-1}$ であることが解る。

ここで以下の補題を考えるわ。
$A^{k+1} = (QQ_1 \cdot\cdot Q_k)(R_k R_{k-1} \cdot\cdot R)$
証明は意外とシンプルよ
$A^{k+1} = (QQ_1 \cdot\cdot Q_k)(R_k R_{k-1} \cdot\cdot R) = (QQ_1 \cdot\cdot Q_{k-1})(Q_k R_k)(R_{k-1} \cdot\cdot R)$
$= (Q_0 Q_1 \cdot\cdot Q_{k-1}) A_k (R_{k-1} \cdot\cdot R) = A(QQ_1 \cdot\cdot Q_{k-1})(R_{k-1} \cdot\cdot R)$
これを繰り返せば A^{k+1} になることが解るわ。

お次は、A^{k+1} の分解と行くぜ。A を対角化させる行列 M を用いて
$M^{-1} A^{k+1} M = D^{k+1} \quad D = diag(\lambda_0, \lambda_1, \cdot\cdot, \lambda_n)$[*4]
とすることができるのは線形代数でおなじみだな。λ は行列 A の固有値で、添え字が小さいほうが絶対値が大きいとしてくれ。

$A^{k+1} = M D^{k+1} M^{-1}$
M^{-1} を **LU 分解**して $L_M U_M$ とすれば
$A^{k+1} = M(D^{k+1} L_M D^{-(k+1)}) D^{k+1} U_M$
L, D の形から $(D^{k+1} L_M D^{-(k+1)}) \equiv C$ とすると

$c_{ij} = l_{ij} (\frac{\lambda_i}{\lambda_j})^{k+1}$
となる。L の形から解るとおり C は下三角行列だ。ここから、C は k が大きくなるにつれて単位行列に収束していくことがわかるだろう。

C が十分単位行列として扱えるとした時、A^{k+1} の式は
$A^{k+1} = MCD^{k+1} U \fallingdotseq MD^{k+1} U_M$
U が上三角行列であることから解るとおり、C をほぼ単位行列とみなせば A_{k+1} が M と $D^{k+1} U_M$ で **QR 分解出来ている**わ。

[*4] $diag(a, b \cdot\cdot)$ は非対角要素が全て 0 で、対角部分が $a_{11} = a, a_{22} = b$ となっていく行列のこと

さて、最初に導出した式
$A^{k+1} = (QQ_1 \cdot \cdot Q_k)(R_k R_{k-1} \cdot \cdot R)$ も見事な A^{k+1} の QR 分解になっているわね。
双方とも QR 分解になっていることから、
$(QQ_1 \cdot \cdot Q_k) = M\Omega$
になるわ。Ω は直行成分に影響をもたらさず、$\Omega^{-1}R$ が上三角行列になるような直行行列よ。Ω^{-1} も直行成分に影響をもたらさないわ。

さて、言うまでもなく最後にやるべきことは A_{k+1} の計算ね
$A_{k+1} = (QQ_1 \cdot \cdot Q_k)^T A (QQ_1 \cdot \cdot Q_k) = \Omega^{-1} M^{-1} A M \Omega = \Omega^{-1} diag(\lambda_0, \lambda_1 \cdot \cdot)\Omega = diag(\lambda_0, \lambda_1 \cdot \cdot)$
こうして QR 分解を繰り返せば直行行列になってくれることが解るわね。固有ベクトルも直行行列で逆変換すれば求められるわね。収束の速さは C の近似を見れば解るように、固有値の比率に依存しているわ。

漸く準備が終わったぜ。さて、はじめるとすっか・・・

12　纏め

連立方程式については纏めるとこんな感じ。

	掃きだし法	LU 分解	ガウス・ザイテル法	共役勾配法
相手となる行列	なんでもいい	なんでもいい	なんでもいい	対称かつ固有値が全て正
解	厳密解	厳密解	近似解	近似解も厳密解もいける
下準備など	ピボット	特には	対角要素を大きく	対角要素を大きく
速さ	遅い、一定	早い、一定	固有値に依存	早いらしい
しけ４の個人評価	吐きだし法	オススメ	覚えておけ	オススメ

固有値問題については纏めるとこんな感じ。計算方法を熟知すればわかるとおり、疎行列に対抗できるのは行列の形にもよるけどランチョス法だけだと思うのでランチョス法はしっかり勉強すること。こと。

	2 分法	ランチョス法	緩和法	QR 法
相手となる行列	対称行列	対称行列	対称行列	対称行列
必要な下準備	ハウスホルダー変換	特になし	特になし	ハウスホルダー変換
固有値の出方	N 個同時に出る	M 個に調節可能	基底状態から順に出る	N 個同時に出る
固有ベクトル	出てこない	出る	出る	出る
しけ 4 の個人評価	ランチョス法に必須	必須技術	これも欲しい	ランチョス法に必須

ぴんぽーん。「突撃！隣の晩御飯」で〜す。

番組枠が圧迫されたから急遽入れたネタじゃないですよ。初めから計画されていたネタですよ？　ですよ？

ぬぬぬ。人が色々忙しいときに困った人ね。
適当にあしらって帰ってもらうわ。それと、魔法使いはうちに居ることばれると不味いからどこかに隠れて・・・

あらあら大変。メインディッシュが逃げようとしてるわよ。
（魔法使いを捕まえて）

隙間を使って勝手に人の家に入るなと何度言ったら解るのかしら。それに、私はそんなに悪食じゃないわ。

じゃあこの縄は何？
もしかして趣味？

じと〜〜〜〜〜〜〜〜〜〜

え、これ、みんな見てるよな・・・・
えと、その、あれだな・・・・・
は、ははははは・・・・

どっきり

へ？

へ？

いやーねー。流石にカメラは回してないわよ。勝手に人の家に上がりこんで撮影なんてしたら犯罪じゃない？
どう、びっくりしたでしょ？
あ、ちなみに、今の時間は「スーパーゲーマーてるよ」による最速 RPG クリアを放送してるわよ。

（放心）

イタイイタイイタイイタイ。
なにかよからぬことを企んでいたからってアンクルホールドかけないで欲しいわね。
4 割くらいおすそ分けくれるなら私も協力するわよ。

（アンクルホールドを解除しながら）
3 割ね。

こうして・・・

（バリトン声で）
こら、人質。もっと苦しそうに喋らんかい！

ぎゃあああああああああああ

（茶を啜りながら）
いやー。アンタ手馴れてるわねえ

がくがくがくがくがくがくがくがく・・・・・

こうしてすべてのツケを無効化に成功した巫女と隙間妖怪なのでした。
得られた資金を元に神社や会社を改築するお話「快的ビフォーアフター」はまたの機会にでも・・・・・

あ、魔法使いも一応ツケを無効化させることには成功しました。
が、彼女には眼鏡の店以外にもツケがあったので、思いのほか生活は楽にならなかったと言います。

【業務連絡】
摩擦班の人に物理教えてもらわないとどうにも手がまわりそうも無いなあ
あと、暇な人群論一緒に勉強しようよ。よ。
ついでにボドゲも遊べると嬉しいなあ。