2010年2月14日

Mathematicaはもういらない

大坂 洋[*1]

[*1] 富山大学経済学部

本論文の目的

- **経済学のツールとしてのMathematicaへの代替手段としてRを提案する.**
  - Mathematicaの欠点とRの長所
  - **代替手段があることすら考慮されない.**

Mathematicaの値段は高過ぎ

- 学生のパソコンに入れてもらえない．
- 卒業後に，使える環境がある保証がない．

RはGNUライセンスのオープンソースソフトウェア．

フリーは品質が悪いか？

NO!!!

- 理工系のソフトウェアはパワーユーザが開発者
  - バグの修正が早い
  - 最もよいアルゴリズム，コードがすぐに反映される．
- 平均的ユーザと開発者の差が大きいと低品質．
  - Excel, Wordのような Office製品．(Open Office.orgも同様)
  - WindowsとLinux．

マクロ経済学IIで必要だったプログラミングの技術

- 変数の理解
- if(条件分岐)とroop(繰り返し計算)
- 重ね合わせ表示を含めたグラフの表示

○ これらを簡単にできるのが，よい道具．
○ さらに，乱数，微分方程式，ニュートン法があれば．(計量なら，Rのほうが…)
⇒ Mathematica(バカ高いが)もRも十分．

LISP って何(Mathematica の起源)

- **なんでも関数**
- `2 + 1`
  - Lisp **の場合**:$\Longrightarrow$ `(+ 1 2)`
  - Mathematica **の場合**: `Plus[1,2]`

○ **要するに** Mathematica **は** Lisp **における** `(func a b)` **を** `func[a,b]` **に置き換えてたもの．** Lisp **の方言．**

○ $2+2$ **を** Mathematica **でそのまま書けるのは，みかけだけ．(シンタックス・シュガー)**

if も関数

C 風の言語

```
if(a>10){//もし10以上なら
puts("a is larger than 10)//印字し,
}else{//そうでなければ,
puts("a is not larger than 10)} //印 字する.
```

$\Longrightarrow$ Mathematica

```
IF[ a > 10, (*もし，10以上なら*)
Print "a is larger than 10.",(*10以上の
場合*)
Print "a is not larger than 10."](*10
未満の場合*)
```

- C**風の言語は，**if, else, (), **など，「接続詞」，「助詞」が豊富．**
- Mathematica **は** if, [], **カンマだけ．**

○ Mathematica **は接続詞すら不足で，助詞のない英語の人にもよみづらい** (**推測**)

ifも関数(もっといやんなる)

C風の言語

```
if(a>10){
puts("a is larger than 10")
}elseif(a>5){
puts("a is not larger than 5")
}elseif(a>0){
puts(" a is plus.")
}else{puts("a is minus.")}
```

$\Longrightarrow$ Mathematica

```
If[a>10,print["a is larger than 10."]
,If[a>5,
Print["a is larger than 5."]
,If[a>0,
  Print["a is plus."],
Print["a is minus."]]]]
```

- なんことやらわかりません．
- そのうえ，カッコ([　])の数があわないと動かない

ループも関数

```
for(i=1;i<10;i++){puts(i)}
```

```
For[i=1,i<10,i=i+1,Print[i]]
```

3**重ループなどやってられん**!!

まるはだかLisp方言としてのMathematicaの短所

あくまで，素人にとってです．

- カッコをコピーし忘れたり，見落したりするだけで動かない．
- ループや条件分岐といった構造がわかりにくい．
- ループや条件分岐がわかりやすく，Lispの利点をもつ言語(内部的には関数言語)もある．

$\Rightarrow$ Ruby, Python, **そして，R!!!**

これらが，みな素人向け言語として評判がいいことに注意．

Lisp主義者への反論

- **ふつうのやつらの上をいくんなら，**Ruby**，**Python**でなく (**C**や** Java**など外道)，はだかの** Lisp(Mathematica**も可) をつかえ!!**

⇑ **コンピュータでふつうのやつらの上にいっても，経済学の先生として，ふつうのやつらの下に行く．(暇人じゃねーよ)**

  - **職業プログラマ向きの言語でも，はだか**Lisp**でない関数型言語が大人気**
  - **理論のプログラム言語は**Basic**でも可．**

グラフの作成

Mathematica: **高機能プロット命令で，なんでも書いちゃおう．**

R: **単機能のプロット命令はキャンバス．それにいろいろ書き加えよう．**

○ **情報処理センターの紀要で，『Rは重ね合せがやりにくい』と書いたのはまちがい．**

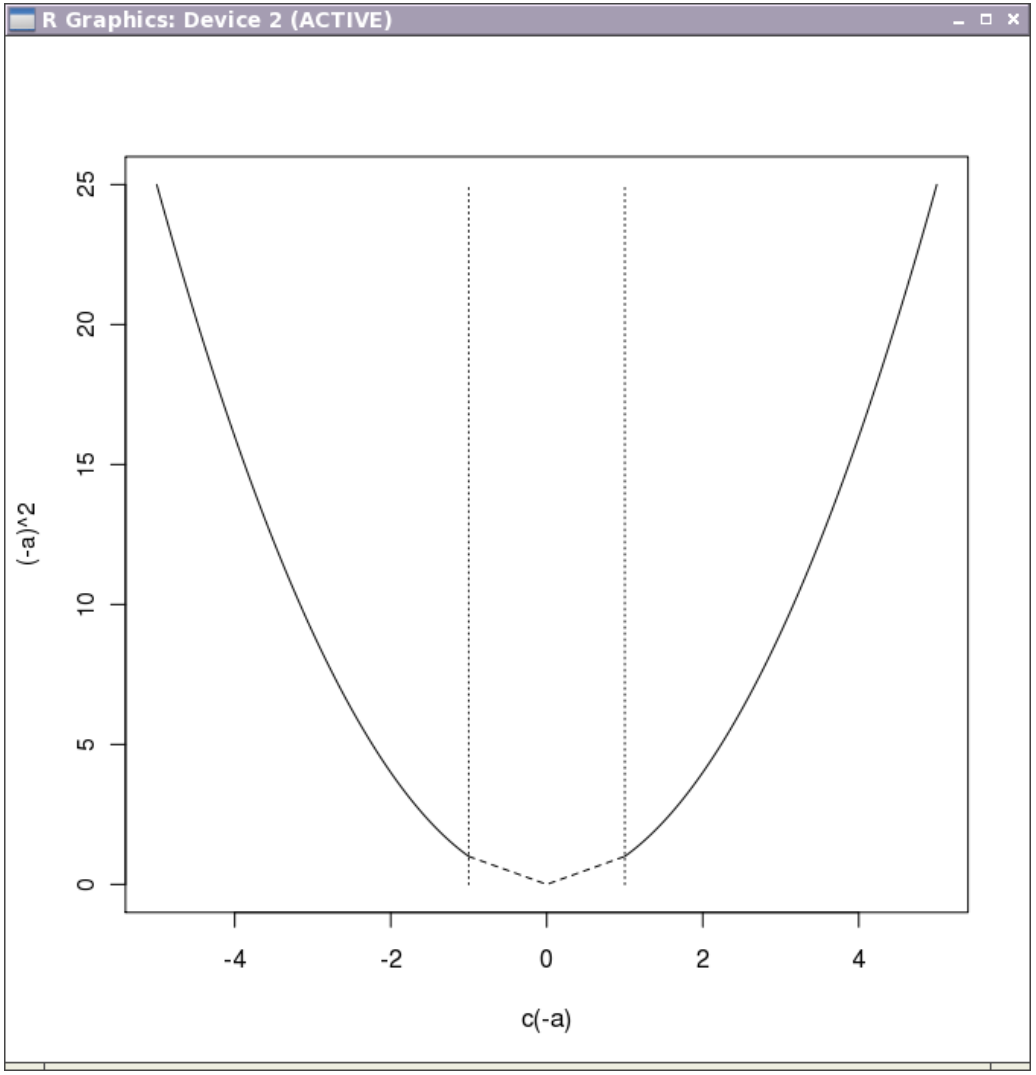
Mathematicaで書きにくくて，Rですぐ書けるグラフの例

$$x < -1, 1 < x \implies f(x) = x^2$$

$$-1 \leq x \leq 0 \implies f(x) = |x|$$

なおかつ，xが$(-\infty,1)$, $(1,0)$,$(0,1)$,$(1,\infty)$で，グラフの線種を変えて，領域の境界に縦線をいれたい．

完成図



実際にしてみる．

グラフの領域は，x**は**$[-5, 5]$**,** y**は**$[0, 25]$**とする．**
1から5**の**0.01**きざみのベクトルを作る．**

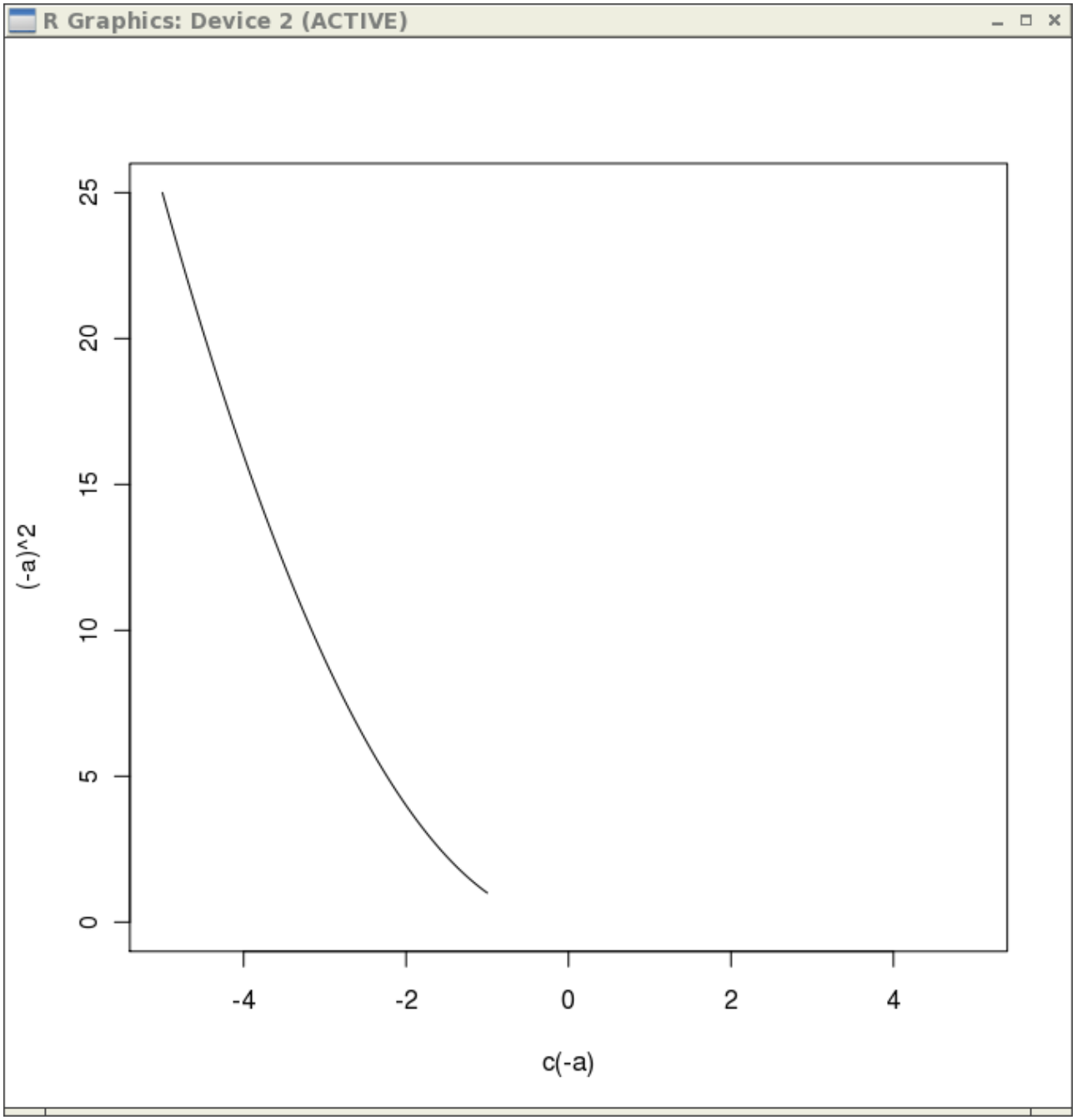
```
>a=seq(1,5,0.01)
```

グラフの領域を指定してplot.

```
plot(-a,(-a)^2,type="l",
xlim=c(-5,5),ylim=c(0,25))
```

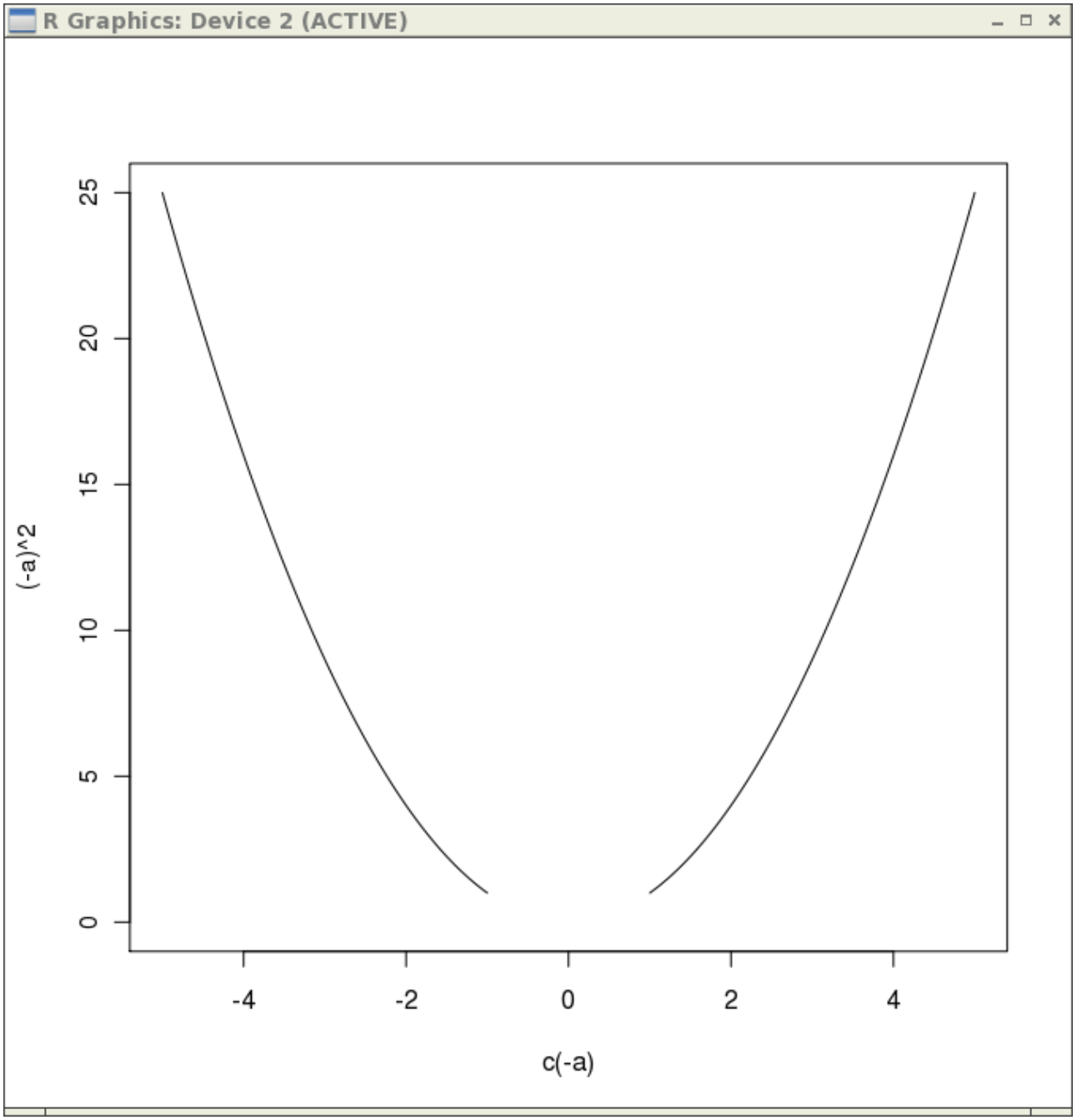
まず，$x = [-5, 1]$ で x^2 のグラフを作る．

```
plot(-a,(-a)^2,type="l",
xlim=c(-5,5),ylim=c(0,25))
```


R Graphics: Device 2 (ACTIVE)
(-a)^2
0
5
10
15
20
25
-4
-2
0
2
4
c(-a)

$x = [1, 5]$のx^2を追加(lines命令)

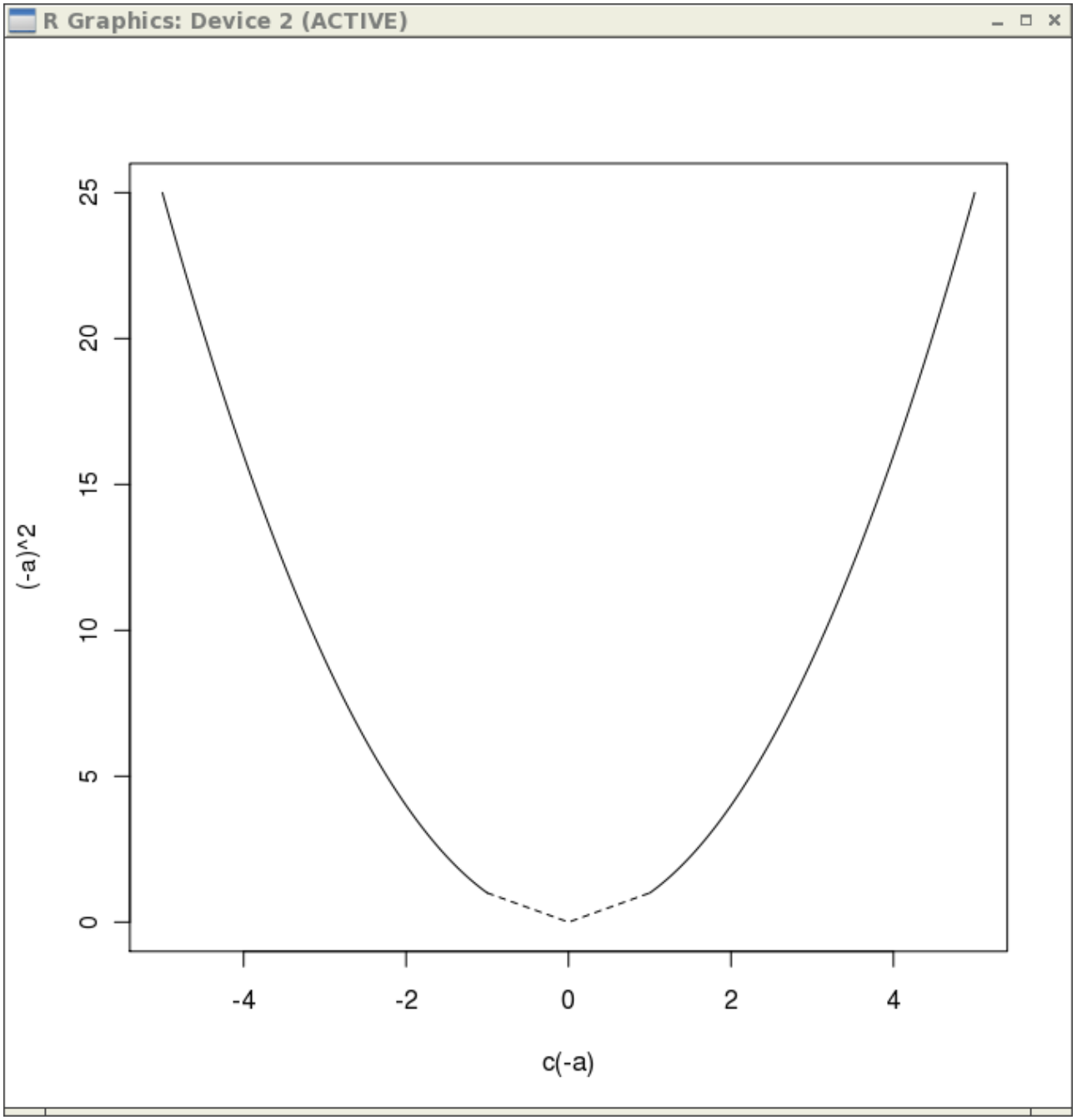
```
> lines(a,a^2)
```


R Graphics: Device 2 (ACTIVE)
(-a)^2
0
5
10
15
20
25
-4
-2
0
2
4
c(-a)

$x = [-1, 1]$**の**$|x|$**を線種をかえてする．**

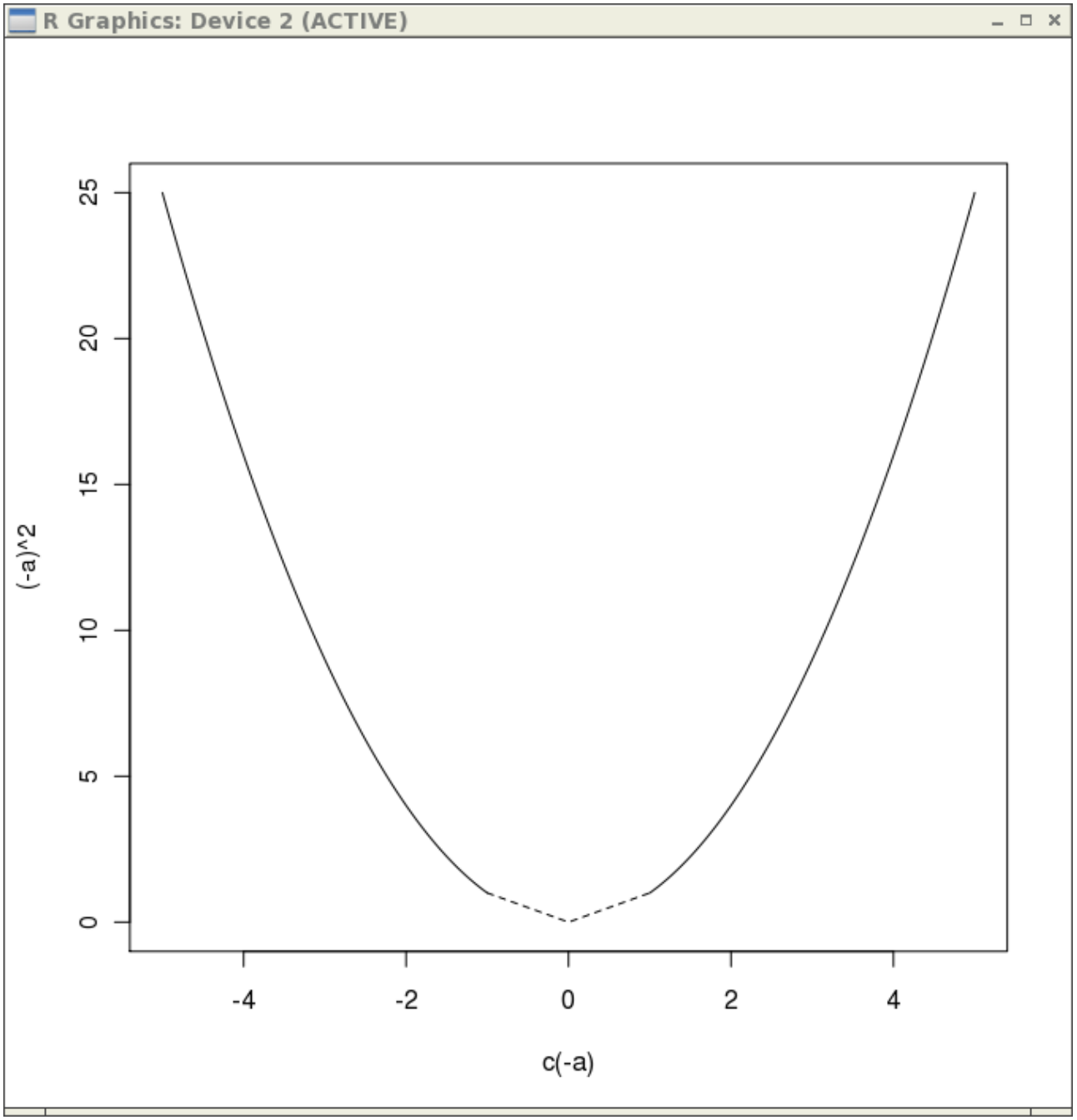
```
> lines(b,abs(b),lty=2)
```

R では，`lty=1, 2, 3...`**とすると適当に線種を変えてくれる．**


R Graphics: Device 2 (ACTIVE)
(-a)^2
0
5
10
15
20
25
-4
-2
0
2
4
c(-a)

領域の境界線を引く．直線$(-1,0)$—$(-1,25)$**と**
$(1,0)$—$(1,25)$

```
> lines(c(-1,-1),c(0,25),lty=3)
> lines(c(1,1),c(0,25),lty=3)
```


R Graphics: Device 2 (ACTIVE)
(-a)^2
0
5
10
15
20
25
-4
-2
0
2
4
c(-a)

経済での応用:蜘蛛の巣過程

石橋他の蜘蛛の巣過程をRでつくってみる．
ここでする方法のほうが，ストレート
Mathematicaでもできるけど....

蜘蛛の巣過程の復習

参考: 西村和雄『ミクロ経済学入門』

- **t期における価格はt期の供給量を売り切る価格によってきまる.**

  $\Rightarrow$ p(t)=invD(s(t))

- **t+1期における価格はt期の価格に対する供給量として決る.**

  $\Rightarrow$ s(t+1)=S(p(t))

『はじめよう経済学のための Mathematica**』**
(pp.170–177**) と同じ設定をする．**

- **需要関数** $D(t) = -200p(t) + 4500$
  $\Rightarrow\ invD(d(t)) = 22.5 - 0.005d(t)$
- **供給関数** $s(t) = 250p(t)$

これより，

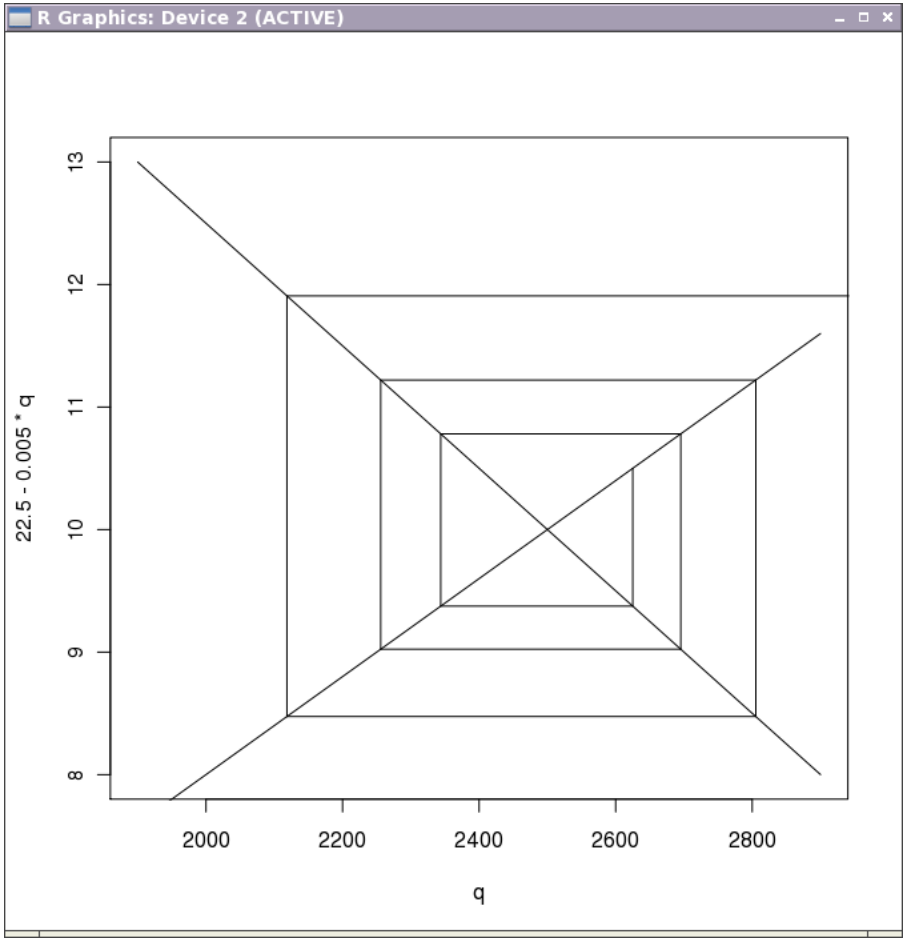
$$\begin{cases} p(t+1) = 22.5 - 0.005s(t) \\ s(t+1) = 250p(t) \end{cases}$$

解を求める

```
> p=NULL
> s=NULL
> p[1]=10.5
> s[1]=250*p[1]
> for(t in 1:6){
+ p[t+1]<-22.5-0.005*s[t]
+ s[t+1]<-250*p[t+1]
+ }
```

グラフをつくる

完成図


R Graphics: Device 2 (ACTIVE)
22.5 - 0.005 * q
13
12
11
10
9
8
2000
2200
2400
2600
2800
q

$(s(t), p(t))$—$(s(t), d(s(t)) = p(t+1))$—
$s(t+1), p(t+1)$ **と線をつないで**
いく．

```
> q<-1900:2900
> plot(q,22.5-0.005*q,type="l")
> lines(q,0.004*q)
> for(t in 1:6){
+ lines(c(s[t],s[t],s[t+1]),
+ c(p[t],p[t+1],p[t+1]))
+ }
```

明らかにここでのやりかたが楽.

なぜ，簡単にやれたか

- 『はじめよう』のスタイルはウルフラム推奨方式
  - 関数プログラミング，ベクトル演算の多用

  $\Rightarrow$ 経済学の説明より，Mathematicaの説明に紙面が占めらがち．
- ここでのやりかたは繰り返し計算のみ
  - 繰り返し計算の書き方の知識だけで十分

  $\Rightarrow$ 経済学の説明に紙面をさける．

Mathematicaでも同じやりかたはできるけど...

- MapとShowの併用

  ⇒ **これだけのためにMapを教えるべきか？**

  ⇒ **構文が関数型プログラミングに特化**

ふたたびRとMathematicaの比較

- 言語に習熟していれば，どちらも同じことができる．(内部的には，RもMathematicaも関数型言語)
- Rは繰り返しの構文の多用を想定しているが，Mathematicaは関数型プログラミングを推奨．
- 経済学のような変数が時間とともに変化するものこそ，繰り返し計算の有利な分野(関数型プログラミングの経典"SCIP"でも認めている．)

"We ordinarily view the world as populated by independent objects, each of which has a state that changes over time. An object is said to "have state" if its behavior is influenced by its history. A bank account, for example, has state in that the answer to the question "Can I withdraw $100?" depends upon the history of deposit and withdrawal transactions."
"Structure and Interpretation of Computer Programs, second edition" by Harold Abelson and Gerald Jay Sussman with Julie Sussman, '3.1 Assignment and Local State'.

(和訳) 我々は世界を独立したオブジェクトが住む世界として見る．それぞれのオブジェクトは時間をつうじて変化する．その振舞がその歴史に影響されるとき，オブジェクトは「状態を持つ」と呼ばれる．たとえば，銀行口座は状態をもつ．というのは，「私は100ドル引き出せるだろうか」という問いへの答えは預金と引き出しの歴史に依存するからである．

MathematicaもExcelよりはまし

道具の数と作業の自由度

1. すべて，一つの道具で．(Excel, Word)
2. 複数の道具の組合せ(UNIXシェル，関数の豊富なプログラミング言語)
3. 道具をつくれる道具で，必要な道具から作る．(ライブラリのなしのC言語，アセンブラ)

○ 1は用途が広がると道具が複雑に．

○ 3は自由度が高いが，時間と熟練が必要．

Mathematicaがダメな理由

- forとかifとかが，書きにくく，読みづらい．括弧の対応を気にしなくてはらない．
- 関数がなんでもできる超高レベル関数とプログラマ向きの超低レベル関数ばっかり．
- 上の二つの複合によって，小さな道具を組合せるようなプログラミングがやりにくい．
  - 道具のつながりがわかりにくい．
  - 高レベルの関数はこまわりのきく道具としては，複雑すぎ．

べつにRでなくてもよい

- 普及しているほとんどのプログラミング言語はどれもMathematicaより習得が簡単で実数の計算能力は十分．(Ruby，Pythonあたりがおすすめ．)
- グラフの作成が汎用のプログラミング言語での問題点．だけど，Gnu Plotのようなグラフ作成ソフトと組合せる手もある．
- 上の手間をさけれる点とベクトルの扱いにRをふくめた，数値解析言語が有利

要するに繰り返し計算しかできない素人が
数値をいじるだけなら，
Mathematica**はクソ!!!**

まとめ

- **石橋先生と険悪なわけではありません.**
- Scheme**万歳!!(理学部**K**先生への配慮**)
- Mathematica**は値段高過ぎ.**
- Mathematica**は素人の数値解析の道具としては，使いにくい.**
- R**以外にもいい道具はいっぱいある.**
- **数式処理には，**Mathematica**はよい.でも，追加情報も参照.**

追加情報:数式処理環境が欲しい人のために

Mathematica以外にも数式処理環境は数多くあり，いくつかのものはフリーソフトです．

- Reduce(いまではフリーです．GPLで公開．)
- Maxima(Mathematicaより，由緒正しい数式処理言語)