



PSTricks 좌표의 이해와 활용 : 경제학 모형의 함수 플로팅을 중심으로

Understanding and Using Coordinates in PSTricks with
Application to Plotting Functions from Economics Models

조인성 In-Sung Cho

공주대학교 경제통상학부 ischo@ktug.or.kr

KEYWORDS PSTricks, coordinate system, plotting functions, economics model

ABSTRACT The article examines the coordinate systems in PSTricks. Though the default is the Cartesian coordinate system, one can also use the polar coordinate system once the command \SpecialCoor is declared. The command \SpecialCoor allows greater variety of expression of the Cartesian and polar coordinates, such as PostScript language and predefined nodes. Understanding the features of the various expression of coordinates would make the tedious graphing job interesting or challenging. The article provides some examples of plotting economics models: markets, monopoly, and Cournot duopoly.

1 머리말

함수를 플로팅하는 데에는 함수 자체를 플로팅하는 것 외에도 다양한 정보를 함께 표시하는 것이 필요하다. 예를 들어 좌표축, 좌표축의 이름, 특정한 점, 점의 이름, 점과 점사이의 보조선 등 다양한 정보가 필요하다. 이러한 정보는 적절한 위치에 놓여야 한다.

함수의 플로팅에 관련된 여러가지 정보를 적절한 위치에 놓기 위해서는 좌표체계에 대한 이해가 반드시 필요하다. 이 글의 목적은 PSTricks의 좌표체계와 좌표의 표현 방법을 이해하고 좌표의 다양한 특수표현을 함수의 플로팅에 활용하는 문제를 살펴보는 것이다.

PSTricks에서 기본으로 사용하는 좌표체계는 데카르트 좌표(Cartesian coordinate) 체계이다. \SpecialCoor를 선언하면 극좌표(polar coordinate) 체계도 함께 사용할 수 있다.

함수의 플로팅에 필수적인 플로팅 관련 PSTricks 패키지들은 \SpecialCoor를 기본으로 하기 때문에 이를 따로 선언하지 않아도 된다. 패키지 pst-plot은 pstricks를, pstricks-add는 pst-plot과 pst-node를, pst-func는 pstricks-add와 pst-math를 자동검색하여 불러들이므로 pstricks 기본 패키지를 제외하면 \SpecialCoor를 따로 선언하지 않아도 된다.

이 글에 나오는 모든 예제는 pstricks-add 패키지만 불러들여도 컴파일할 수 있다. 대부분의 경우 \usepackage{pst-func}만으로도 큰 불편 없이 함수의 2차원 플로팅 작업을 할

수 있다.¹

PSTricks에서 사용되는 모든 좌표는 둑근 괄호로 감싸서 표현한다. 데카르트 좌표는 (x, y) 와 같이 두 실수를 콤마로 구분하고, 극좌표는 $(r; \theta)$ 와 같이 중심에서의 길이를 나타내는 반지름(radius)과 각도를 세미콜론으로 구분해서 표현한다. \SpecialCoor가 선언되면 극좌표를 쓸 수 있을 뿐 아니라 매우 다양한 좌표의 표현 방법이 허용된다. \SpecialCoor가 받아들이는 다양한 특수 표현을 특수좌표라 부른다.

특수좌표에는 포스트스크립트 언어를 직접 쓸 수 있다. 포스트스크립트 언어는 좌표내에서 함수값을 계산하는 등 매우 강력한 도구로 이용될 수 있다. 0-차원의 노드(node)좌표를 정의하는 매크로 \pnode도 매우 유용하다. \SpecialCoor가 선언되면 각도의 표현에도 포스트스크립트 코드나 노드좌표를 사용할 수 있다.

\pnode를 내부적으로 사용하면서 그 기능을 확장한 다양한 매크로가 개발되어 왔다. 이들 매크로는 pst-node [10], pst-plot [9], pstricks-add [4] 등을 중심으로 소개되어 있다. 이 글에서는 이들 패키지와 pstricks 기본 패키지 [7]에 산재해 있는 좌표의 개념을 정리하고자 한다. 또한 모든 기능을 다 다룰 수는 없으므로 경제학 모형의 플로팅에 유용한 기능을 중심으로 살펴보기로 한다.

이 글을 읽는 데에는 PSTricks에 관한 몇 가지 기초지식이 필요하다. 이 글의 예에서 자주 사용되는 그래픽 명령 몇 가지를 나열하면 다음과 같다.

```
\psdots [par] (coor1)(coor2)...
\psline [par] (coor1)(coor2)...
\psarc [par] (coor){radius}{angle1}{angle2}
\uput{labelsep}[refangle]{rot}(coor){stuff}
\psplot [par] {xmin}{xmax}{function}
```

\psdots는 지정한 좌표에 점(bullet)을 찍어 준다. \psline은 나열된 좌표를 선으로 연결하며 좌표가 하나만 표시되면 원점에서 그 점으로 선을 그려준다. \psarc는 지정한 좌표를 중심으로 angle1부터 angle2까지 반시계방향으로 회전하며 반지름이 radius인 호(arc)를 그려준다. \uput은 지정한 좌표에 라벨을 붙이는 데 사용되는 유용한 매크로이며, \psplot은 함수를 플로팅하는 매크로이다. pspicture 환경이나 \psplot, \grid, \psaxes 등 함수의 플로팅에 관한 기초적인 명령은 [1]에서 논의되었으며, 보다 자세한 논의는 [9], [4], [6], [3] 등에서 찾아볼 수 있다.

이 글의 구성은 다음과 같다. 2절에서는 PSTricks의 기본 좌표체계인 데카르트 좌표체계와 극좌표 체계를 살펴본다. 3절에서는 데카르트 좌표와 극좌표를 표현하는 다양한 특수 표현들에 대해 살펴보고, 이 중에서 노드좌표의 기능을 다양하게 확장시키는 매크로를 4절에서 살펴본다. 5절에서는 몇 가지 경제학 모형을 플로팅하는 문제를 검토하고, 6절에서 결론을 맺는다.

1. 이 글에서 함수의 3차원 플로팅은 다루지 않는다. 함수의 3차원 플로팅에 관해서는 [5]를 보라.

2 PSTricks 좌표 체계

2.1 데카르트 좌표 체계

PSTricks 패키지에서 사용되는 좌표체계는 데카르트 좌표체계를 기본으로 하며, 데카르트 좌표는 (x, y) 와 같이 콤마로 분리된 두 수를 둘째 괄호로 묶어서 나타낸다. 좌표는 $(2\text{cm}, 3\text{pt})$ 와 같이 길이 단위를 명시적으로 표시할 수 있으며, $(2, 3\text{mm})$ 나 $(2, 3)$ 과 같이 숫자만 쓸 수도 있다. 길이 단위가 생략되면 미리 설정된 단위가 사용되며, 따로 설정하지 않았을 경우에 기본값은 1cm 이다. 즉 기본값이 적용될 경우, $(2, 3)$ 과 $(2\text{cm}, 3\text{cm})$ 는 서로 같다.

가로 길이 단위는 `xunit`, 세로단위는 `yunit`, 반지름 단위는 `runit`의 값으로 각각 조절할 수 있으며, 이 세 파라미터를 한꺼번에 조절하려면 `unit=.5cm`와 같이 `unit` 파라미터를 이용한다. 길이 단위를 나타내는 이 파라미터들의 기본값은 1cm 이다. `\psset{unit=.5}`와 같이 길이 단위를 쓰지 않으면 현재 설정된 단위 길이의 0.5배가 새로운 길이 단위로 설정된다.

각의 기본 단위는 `\degrees[num]` 명령에서 숫자 `num`으로 설정하며, 기본값은 도(degree) 단위로 원 전체를 표시하는 `\degrees[360]`이다. 백분율로 표시하는 파이 차트를 그린다면 `\degrees[100]`과 같이 100단위로 설정하는 것이 편리하다. 라디안(radian) 단위로 설정하는 `\degrees[6.28319]` 명령은 간단히 `\radians`로 대체할 수 있다.

옵션 `origin`은 원점을 이동시킬 때 사용한다. 원점을 가로로 x , 세로로 y 만큼 이동시키려면 `origin={x,y}`와 같이 설정한다. 여기서 x 와 y 의 순서쌍이 둘째 괄호가 아니라 중괄호({ })로 묶여 있는 점에 유의하자. 직선의 그래프와 점이 $\{2, 1\}$ 만큼 이동하는 예가 그림 1에 나타나 있다(코드 7째 줄 참고).

```

1 \psset{xunit=.7,yunit=.7}
2 \begin{pspicture}[showgrid](-1,-1)(4.5,4.5)
3 \psset{algebraic}
4 \psaxes{->}(0,0)(-1,-1)(4.5,4.5)
5 \psplot{0}{2}{4-2*x}\psdot(0,4)
6 \psplot[swapaxes=true]{0}{2}{4-2*x}
7 \psset{origin={2,1}} % 원점 이동
8 \psset{linestyle=dashed,ticks=none}
9 \psaxes{->}(0,0)(-1,-1)(4.5,4.5)
10 \psplot{0}{2}{4-2*x}\psdot(0,4)
11 \end{pspicture}
12 \psset{unit=1cm}

```

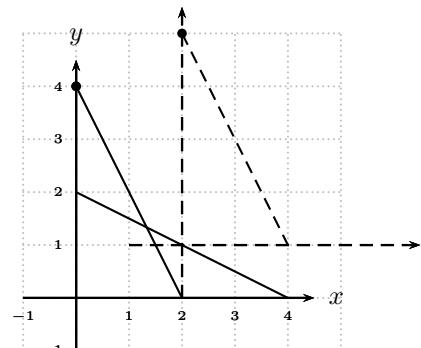


그림 1. `origin` 옵션과 `swapaxes` 옵션

그래픽 개체를 45° 선에 대칭시키려면 옵션 `swapaxes`를 이용하면 간단하다. 그림 1의 코드 6째 줄에서 보듯이, $y = 4 - 2x$ 의 그래프를 `swapaxes` 옵션과 함께 플롯하면, x 와 y 를 바꾼 $x = 4 - 2y$, 즉 $y = 2 - \frac{1}{2}x$ 의 그래프를 그리는 것과 같은 효과를 낼 수 있다.²

2. 옵션 `shift`는 `pspicture` 환경이 만드는 box 자체를 수직으로 올리고 내리는 기능을 가진 것으로, `origin`이나 `swapaxes` 옵션과는 달리 좌표체계에 영향을 주지는 않는다.

2.2 극좌표 체계

PSTricks에서는 데카르트 좌표와 함께 극좌표도 사용할 수 있다. 극좌표는 반지름 r 과 각 a 를 세미콜론으로 구분하여 $(r; a)$ 와 같이 표현한다. pstricks 기본 패키지만 불렀을 때 극좌표를 사용하려면, 전처리부 또는 본문의 필요한 곳에 \SpecialCoor 명령을 선언해야 한다.

데카르트 좌표만 사용하는 경우에 \NormalCoor 명령을 내리면 처리속도를 향상시킬 수 있으나, 컴퓨터 성능이 좋아진 요즈음, 처리속도는 중요한 논점이 아니다[3, p. 296]. 이러한 맥락에서 기본 패키지인 pstricks를 제외한 다른 주요 패키지에서 \SpecialCoor를 기본으로 선언해 두고 있다. 구체적으로 pst-node, pst-plot, pstricks-add, pst-func 패키지 등을 불러 들이면 \SpecialCoor를 따로 선언할 필요가 없다.

pst-plot 패키지를 불렀을 때, 다음과 같이 \psaxes 명령에 axesstyle=polar 옵션을 주면 극좌표의 그리드(grid)를 표시할 수 있다.

```
\psaxes [axesstyle=polar] (xMax,yMax)
```

극좌표 그리드와 극좌표를 그리는 예는 그림 2에 있다. 코드 8째 줄에서 볼 수 있듯이 데카르트 좌표와 극좌표는 섞어서 쓸 수 있다.

```

1 % \usepackage{pst-plot}
2 \begin{pspicture}[showgrid](-3,-3)(3,3)
3 \psset{tickcolor=black!80}
4 \psset{subticklinestyle=dashed,
5      subtickcolor=gray,subticks=2}
6 \psaxes[axesstyle=polar](2,2)
7 \psdots(2;45)(1;210)(1.5;-60)
8 \psline(1;210)(-3,-3)(1.5;-60)
9 \psdot(-3,-3)
10 \end{pspicture}
```

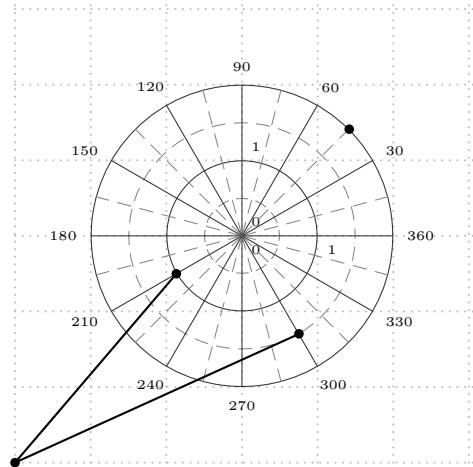


그림 2. 극좌표와 극좌표 그리드

3 PSTricks 좌표의 특수 표현

\SpecialCoor 명령이 받아 들이는 좌표의 표현은 극좌표 이외에도 여러가지가 있다. pstricks 패키지 매뉴얼 [7] 또는 LATEX Graphics Companion [3]에서는 데카르트 좌표만을 보통좌표라 하고 극좌표를 포함하여 특수한 표현을 가진 다른 모든 좌표를 통칭하여 특수좌표라 한다. 엄밀히 말하면, PSTricks에서 사용하는 좌표 체계는 ‘데카르트 좌표’와 ‘극좌표’ 두 가지뿐이며 다른 모든 표현은 이 두 종류의 좌표를 나타내는 특수 표현일 뿐이다. 이 글에서는 [7]을 따라 데카르트 좌표 이외의 다른 모든 형태의 좌표를 특수좌표라 부른다.

이 절에서는 다양한 특수좌표를 살펴보기로 한다. 특수좌표를 사용하려면 `\SpecialCoor` 명령이 먼저 선언되어 있어야 한다는 사실을 기억하자. PSTricks에서 모든 좌표들은 둥근 괄호로 묶어 표시한다.

3.1 여러 가지 특수좌표 표현

매크로 `\pnode`는 “named point”를 정의한다.

```
\pnode{coor}{nodename}
```

즉, `\pnode`는 0-차원의 노드 또는 점을 정의하고 이름을 부여하며, 이렇게 정의된 점이 좌표로서의 기능을 한다. 이를 노드좌표라 한다.

예를 들어, `\pnode(1,1){A}` 명령은 노드좌표 *A*를 (1,1)로 정의한다. 이렇게 정의된 노드좌표 *A*의 위치에 점을 찍으려면, 간단히 `\psdot(A)` 명령을 주면 된다. 노드좌표는 `\pnode(2;30){A}`와 같이 극좌표로도 정의된다.

매크로 `\pnode`가 어떤 위치에 0-차원의 점을 정의하지만, 이곳에 간단하게 점이나 원 등을 찍으려면 이를 그리는 그래픽 매크로를 또 사용해야 한다. 어떤 위치에 직접 점이나 원 또는 상자 등을 놓고 이를 노드좌표로 정의하는 것이 편리할 때도 있다. 이와 관련된 몇 가지 매크로를 소개한다. ([7, pp.60–61]을 참조하라.)

```
\dotnode[par]{coor}{name} % \pnode(2,3){name}\psdot(name)과 동일
\cnode[par]{coor}{name}
\cnodenode[par]{coor}{radius}{name}
\fnode[par]{coor}{name}
```

매크로 `\dotnode{coor}{name}`은 `(coor)`의 위치에 점을 찍고 이름 ‘`name`’을 부여한다. 이는 `\pnode{coor}{name}\psdot(name)`과 동일하다. `\psdot` 매크로로 찍는 점의 기본 크기는 ‘`dotsize=2pt 2`’로 이는 2pt에 `linewidth`의 두 배를 더한 크기이다 [7, p.16]. 매크로 `\cnode{coor}`는 `(coor)`의 위치에 원을 그리고 `name`을 부여한다. 원의 크기는 `radius` 파라미터로 조절하며 기본값은 `.25cm`이다. 매크로 `\cdode`는 `\cnode`와 동일한데 반지름의 크기를 인자로 줄 수 있다. 매크로 `\fnode{coor}`의 위치에 직사각형의 중심을 두고 이름을 부여한다. 직사각형의 가로와 세로는 `framesize` 파라미터로 조절한다. 예를 들어 `framesize=20pt 10pt`는 가로와 세로가 각각 20pt와 10pt인 크기를 나타낸다. 숫자가 하나만 표시되면 정사각형을 그려주며 기본값은 `framesize=10pt`이다. 그림 3에서 예를 볼 수 있다. 각 매크로 명령의 끝에 별표(*)가 붙으면 내부가 태두리의 색으로 채워진 원이나 사각형을 나타낸다.

그림 3의 코드 3–6째 줄은 다음과 같이 `\pnode`를 이용한 일련의 매크로로 바꾸어도 마찬가지 결과를 얻을 수 있다.

```
\pnode(1,1){Dot}\psdot[dotsize=2pt 2](Dot)
\pnode(1,2){Cnode}\pscircle(Cnode){.1cm}
\pnode(2,1){cnodenode}\pscircle(cnodenode){.25cm}
\pnode(2,2){fnode}\psframe*(1.824,1.824)(2.176,2.176) % 2+/- .176
```

```

1 \begin{pspicture}[showgrid](3,3)
2 \psaxes(3,3)
3 \dotnode(1,1){Dot}
4 \Cnode[radius=.1cm](1,2){Cnode}
5 \cnode(2,1){.25cm}{cnode}
6 \fnodes*(2,2){fnode}
7 \fnodes(3;90){polar}
8 \end{pspicture}

```

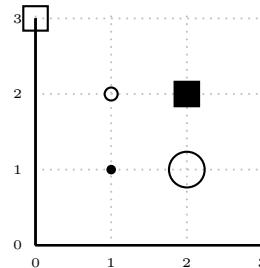


그림 3. 노드좌표

특수좌표 (coor1|coor2)는 (coor1)의 x 좌표와 (coor2)의 y 좌표로 이루어진 새로운 좌표를 정의한다. 이를 혼합(mixed) 좌표라 한다.³ 예를 들면, $(3, 1 | 2, 2.5) = (3, 2.5)$ 이며, 그림 4에서 이들 좌표가 ▲으로 나타나 있다.

혼합좌표를 구성할 때 데카르트 좌표뿐만 아니라 극좌표도 쓸 수 있다. 극좌표를 쓸 경우 $(r; \theta)$ 는 데카르트 좌표의 x -성분 $r \cos \theta$ 와 y -성분 $r \sin \theta$ 로 각각 변환된다. 그림 4에서 코드 6째 줄은 극좌표 $(2; 60)$ 과 원점의 노드좌표 (O)로 이루어진 혼합좌표를 나타내고 있다. 혼합좌표 $(2; 60 | O)$ 은 $(1, 0)$ 으로 x -축에 나타나 있고, 혼합좌표 $(O | 2; 60)$ 은 $(0, \sqrt{3})$ 으로 y -축에 나타나 있다.

```

1 \begin{pspicture}[showgrid](4,3)
2 \psaxes(4,3)
3 \psdots[dotstyle=triangle*](3,1)(2,2.5)
4 \psdot[dotstyle=triangle*](3,1|2,2.5)
5 \psdot(2;60)
6 \psdots(2;60|0)(0|2;60)
7 \end{pspicture}

```

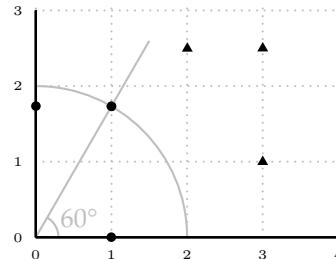


그림 4. 혼합좌표

포스트스크립트 좌표는 강력한 포스트스크립트 언어를 직접 사용하는 특수좌표이며, (!ps)와 같이 ‘!’ 표시로 시작한다. 포스트스크립트 좌표는 구체적으로 (! x-coor y-coor)의 형태를 가지며, 포스트스크립트 언어가 사용하는 RPN (Reverse Polish Notation) 방식으로 쓴다. 예를 들어 대수적 표기로 쓴 ‘ $1 + 2$ ’를 RPN 방식으로 바꾸면 ‘ $1\ 2\ add$ ’와 같다. 표 1은 사칙연산을 RPN으로 표현하는 예를 보여주고 있다. add나 mul과 같은 이항연산 명령들은 두 개의 숫자 인자 다음에 나온다.

대수적 표현 ‘ \sqrt{x} ’와 ‘ $\cos x$ ’는 RPN 방식으로 각각 ‘ $x\ sqrt$ ’와 ‘ $x\ cos$ ’으로 쓴다. 함수의 인자들은 명령 앞에 온다. 이 글에서 나타나는 함수 명령을 포함한 몇 가지 간단한 포스트스크립트 명령을 표 2에서 볼 수 있다. 보다 자세한 포스트스크립트 명령은 [2]를 보라.

3. 특수좌표의 명칭은 pstricks.tex (version 2.15, 2010/11/25)의 755–762째 줄을 참고하였다. Goossens et al. [3, p.298]에서는 복수(doubole) 좌표라 부른다.

표 1. 사칙연산 RPN [1, p.92]

RPN	대수적 표기	의미
1 2 div	$1/2$	$\frac{1}{2}$
x 1 add	$x+1$	$x + 1$
1 2 div x 1 add mul	$(1/2)*(x+1)$	$\frac{1}{2}(x + 1)$
x 1 sub	$x-1$	$x - 1$

표 2. 포스트스크립트 명령

RPN	대수적 표기	의미 / 결과
x sqrt	\sqrt{x}	\sqrt{x}
Euler x exp	$\text{EXP}(x)$ (or Euler^x)	e^x
x cos (degrees)	$\cos(x)$ (radians)	$\cos x$
x atan (degrees)	$\text{atan}(x)$ (radians)	$\arctan x$
/x a def	-	$x = a$
x y exch	-	y x
x dup	-	x x
x dup mul	$x*x$	x^2

Euler는 자연지수 e 의 값으로 정의된 상수이다. 함수 $e^x/(x^2 + 1)$ 을 RPN 방식으로 쓰면 ‘Euler x exp x dup mul 1 add div’와 같다. 옵션 algebraic을 쓰면 이 함수를 ‘Euler^x/(x²⁺¹)’ 또는 ‘EXP(x)/(x²⁺¹)’과 같이 익숙한 대수적 표기로 쓸 수 있다. 그림 6의 코드 8째 줄이 옵션 algebraic의 사용 예이다. 함수의 플로팅과 algebraic 옵션에 관한 기초적인 논의는 [1]을 보라.

좌표 $(x + 1, 2x)$ 를 포스트스크립트 좌표로 쓰면 (! x 1 add 2 x mul)과 같다. 극좌표 $(r; a)$ 의 x -성분은 $r \cos a$ 이고 y -성분은 $r \sin a$ 이므로 이를 포스트스크립트 좌표로 표현하면 (! r a cos mul r a sin mul)이다. 그림 5는 포스트스크립트 좌표의 예를 보여주고 있다. 코드 3째 줄은 $A = (2, \sqrt{3})$, 코드 5째 줄은 $B = (3, 3 - 1) = (3, 2)$ 를 의미한다. 또한 코드 10째 줄은 $C = (4; 60)$ 을 뜻한다.

대수적 포스트스크립트 좌표는 그래프 상의 한점 $(x, f(x))$ 을 편리하게 나타낼 수 있도록 $(*{x} {f(x)})$ 의 형태를 갖는 특수좌표로, ‘!’ 표시 대신에 ‘*’ 표시로 시작한다. 대수적 포스트스크립트 좌표의 x -성분은 RPN 방식으로, y -성분인 함수값 ${f(x)}$ 는 대수적 표기로 써야 하며, ‘*’ 표시와 이어지는 ${x}$ 사이에 공백을 두지 않도록 주의해야 한다. 어느 성분이든 단순히 숫자나 문자 하나만으로 이루어지지 않은 경우에는 항상 중괄호 {}로 묶어야 한다. 포스트스크립트에서는 각의 단위가 도(degree)이지만 대수적 표현에서는 라디안(radian)이라는 점도 유의해야 한다.

그림 6에서 코드 5째 줄은 좌표 (A) 를 극좌표 $(2; 60)$ 로 정의하며, 코드 9째 줄은 좌표 (B) 를 $f(x) = \frac{1}{10}e^x$ 일 때 그래프 상의 점 $(2.5, f(2.5))$ 로 정의한다.

```

1 \begin{pspicture}[showgrid](4,4)
2 \psaxes{->}(4.5,4.5)
3 \pnodex{!2 3 sqrt}{A}
4 \psdot(A)\uput[u](A){$A$}
5 \pnodex{!3 dup 1 sub}{B}
6 \psdot(B)\uput[u](B){$B$}
7 \psdot(!2 60 cos mul 2 60 sin mul)
8 \psarc(0){2}{!0 10 add}{!90 10 sub}
9 \psarc(0){4}{!0 10 add}{!90 10 sub}
10 \pnodex{!4 60 cos mul 4 60 sin mul}{C}
11 \uput[u](!4 60 cos mul 4 60 sin mul){$C$}
12 \psline(C)\psdot(C)
13 \end{pspicture}

```

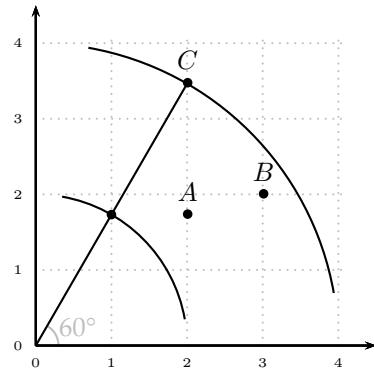


그림 5. 포스트스크립트 좌표

```

1 \begin{pspicture}[showgrid](4,4)
2 \psaxes{->}(4,4)
3 \psdots(*2 x)(*3 {sqrt(x)+1})
4 \psset{dotstyle=o}
5 \pnodex{!2 60 cos mul} {2*sin(\psPi/3)}{A}
6 \psdot(A)\uput[d](A){$A$}
7 \psarc(0){2}{-5}{95}
8 \psplot[algebraic]{-.5}{3.5}{0.1*EXP(x)}
9 \pnodex{2.5 {0.1*EXP(x)}}{B}
10 \psdot(B)\uput[d](B){$B$}
11 \psline[linestyle=dotted](0|B)(B)(B|0)
12 \end{pspicture}

```

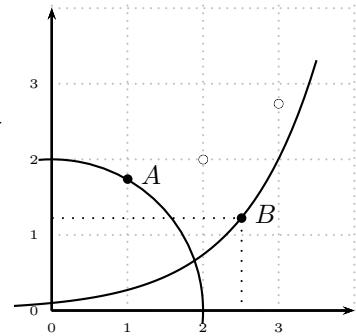


그림 6. 대수적 포스트스크립트 좌표

특수노드좌표는 임의의 노드좌표를 중심으로 상대적인 위치를 정의하는 특수좌표이다.⁴ 상대적인 위치는 `nodesep`, `offset`, `angle`, 세 개의 파라미터와 그 조합으로 조절한다. 수평 이동은 `nodesep`, 수직 이동은 `offset`으로 조절하며, `angle`은 이동한 점을 회전시킨다.

예를 들어 $A = (2, 2)$ 일 때 (`[nodesep=2, offset=-1] A`)는 A 로부터 오른쪽으로 2단위, 아래로 1단위 이동한 좌표 $(4, 1)$ 을 의미하며, (`[nodesep=1, angle=45] A`)는 A 를 중심으로 오른쪽으로 1단위 이동한 후 45° 만큼 회전한 위치를 의미한다. 이러한 예들이 그림 7에서 좌표 A, B, C 로 나타나 있다.

어떤 좌표를 중심으로 상대적인 좌표를 반복해서 사용한다면, 그림 7의 코드 5–6째 줄의 `\RELpnodex`와 같이 임시 매크로를 정의해서 사용하면 편리하다. 그림 7의 코드 7째 줄의 `\RELpnodex{A}{2}{-1}{B}`는 A 에서 가로축으로 2, 세로축으로 -1 만큼 이동한 점을 B 로 정의한다.

4. `pstricks` 패키지 매뉴얼 [7]에서 언급된 또 다른 특수노드좌표 (`[par]{nodeB}nodeA`)는 점 A 를 중심으로 B 와의 연장선 상의 상대적 위치를 정의한다. 이는 활용 빈도가 떨어질 뿐 아니라 다른 편리한 명령으로 대체될 수 있으므로 이 글에서는 자세히 언급하지 않는다.

```

1 \begin{pspicture}[showgrid](4,4)
2 \psaxes(4,4)
3 \pnode(2,2){A}\uput[100](A){$A\$}
4 \psdots(A)([offset=1]A)([nodesep=-1]A)
5 \def\RELpnode#1#2#3#4{%
6   \pnode([nodesep=#2,offset=#3]#1){#4}}
7 \RELpnode{A}{2}{-1}{B}
8 \psline(A)(B)
9 \psdot(B)\uput[100](B){$B\$}
10 \pnode[nodesep=1,angle=45]A{C}
11 \psdot(C)\uput[100](C){$C\$}
12 \end{pspicture}

```

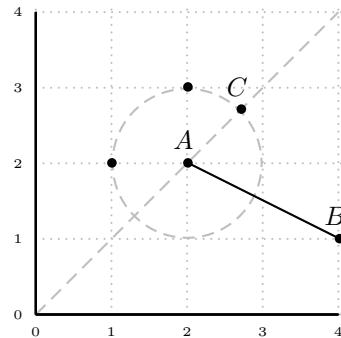


그림 7. 특수노드좌표

3.2 각의 특수 표현

각의 단위인 도는 보통 숫자로 표현하지만, \SpecialCoor가 선언되면 좌표 (coor)나 !ps, 즉 포스트스크립트 코드로 표현할 수 있다.

원점에서 y -축에 놓여 있지 않은 점 $A = (x, y)$ 의 방향으로 뻗어나는 선과 가로축이 이루는 각을 θ 라 하면 $\tan \theta = \frac{y}{x}$ 이며, 역으로 $\theta = \arctan \frac{y}{x}$ 이다. 이 경우, 극좌표 $(r; \theta)$ 를 나타내기 위해서 $(r; \{(x, y)\})$ 나 $(r; \{(A)\})$ 와 같이 각도의 표현에 좌표를 쓸 수 있다.⁵ 또한, ‘ $\arctan \frac{y}{x}$ ’를 포스트스크립트의 RPN 방식으로 표현하면 ‘ $y\ x\ atan$ ’이므로, $(r; !y\ x\ atan)$ 과 같이 ‘!’ 표시에 이어지는 포스트스크립트 코드로 각도를 표현할 수도 있다.

그림 8의 예에서 원점 O 와 점 $A = (2, 3)$ 를 연결하는 선과 가로축이 이루는 각을 θ 라 하면 $\theta = \arctan \frac{3}{2} \approx 56.3099^\circ$ 이다.⁶ 코드 5째 줄은 원점에서 $(2, 3)$ 의 방향으로 길이가 4.5cm인 선을 그리라는 것이며, 코드 6째 줄은 원점을 중심으로 하고 반지름이 4cm인 호를 0° 에서 $(2, 3)$ 방향(즉 56.3099°)까지 그리라는 것이다. 코드 7째 줄은 원점을 중심으로 하고 반지름이 3cm인 호를 $(2, 3)$ 방향에서 시계방향으로 45° 만큼 작은 각도(즉 11.3099°)까지 그리라는 것이다. 코드 13째 줄은 점 A 를 중심으로 오른쪽으로 2단위 이동한 후 $180^\circ - 30^\circ = 150^\circ$ 만큼 회전시킨 점까지 점선을 그리라는 것이다.

4 노드좌표의 확장

PSTricks 패키지는 1990년대 초반 Timothy Van Zandt가 개발한 이후 계속해서 그 기능이 확장되며 빠르게 변화하고 있다.⁷ pst-node 패키지의 경우도 마찬가지이다.

노드는 PSTricks에서 가장 강력한 특징 중 하나이다.⁸ 이 절에서는 pst-node 패키지 매뉴

5. pstricks.tex 현재 버전(v. 2.15, 2010/11/25)에서는 극좌표를 표현할 때 {(coor)}와 같이 좌표를 중괄호 {}로 다시 묶어야 한다.

6. 56.3099° 는 \psPrintValue{3 2 atan}의 결과값이다. \psPrintValue를 쓰려면 pst-func를 불러야 한다.

7. PSTricks 프로젝트의 관리는 Denis Girou를 거쳐 현재 Herbert Voß에 의해 이루어지고 있다.

8. 위치를 노드좌표로 정의하고 노드에 라벨을 붙이고 노드를 연결하는 등 노드와 관련된 여러 매크로에 대해 개발자인 Timothy Van Zandt는 “...They are the trickiest tricks!” 라 쓰고 있다 [7, p. 58].

```

1 \begin{pspicture}[showgrid](4,4)
2 \psaxes{->}(0,0)(-.5,-.5)(4.5,4.5)
3 \pnode(2,3){A}\uput[200](A){$A\$}
4 \psdots(0)(A)
5 \psline{->}(4.5;{(A)})
6 \psarc{->}(0){4}{0}{(A)}
7 \psarcn{->}(0){3}{(A)}{!3 2 atan 45 sub}
8 \psline[linewidth=1.2pt](0)(2,0)(2,3)
9 \psarc(0){.3}{0}{(2,3)}
10 \uput{.35}[30](0){$\theta$}
11 \uput[u](1,0){$x$}\uput[l](2,1.5){$y$}
12 \psset{linestyle=dotted,linewidth=1.6pt}
13 \psline(A)[nodesep=2,angle=!180 30 sub]A)
14 \end{pspicture}

```

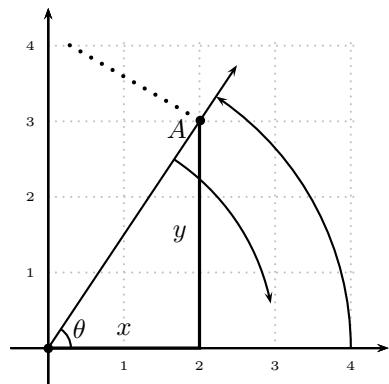


그림 8. 각도의 특수 표현

열 [10]에서 소개된 내용을 중심으로 기능이 확장된 다양한 노드좌표 관련 매크로를 간략히 살펴보기로 한다. 노드좌표를 이해하는 가장 기본적인 매크로는 `\pnode`이며 여기에서 소개되는 다양한 매크로들은 내부에서 `\pnode`를 사용한다.

4.1 선형결합 노드

```

\psLCNode(A){a}{B}{P}
\psLCNodeVar(A)(B)(a,b){P}
\nodexn{linear combination expr}{name} %% example: \nodexn{a(A)+b(B)}{P}
\psLNode(A)(B){b}{P} %% affine combination (a=1-b)

```

두 점 A 와 B 의 선형결합(linear combination) $P = aA + bB$ 는 다음과 같은 세 가지 명령을 통해 노드로 정의할 수 있으며 모두 동일한 결과를 갖는다.

```

\psLCNode(A){a}{B}{P}
\psLCNodeVar(A)(B)(a,b){P}
\nodexn{a(A)+b(B)}{P} % syntax:\nodexn{linear combination}{name}

```

매크로 `\psLCNodeVar`는 `\psLCNode`와 같은 기능을 가지지만, 그림 9의 코드 7째 줄과 같이 `\psLCNodeVar`는 가중치 (a, b) 에 `\SpecialCoor`가 허용하는 특수 표현을 사용할 수 있다는 점과 `\psLCNodeVar(A)(B)(1,1){A}`의 예에서와 같이 노드 A 를 새로운 값으로 덮어 써서 다시 정의하는 것이 허용된다는 점이 다르다.

가중치가 $a + b = 1$ 일 때, `\psLNode(A)(B){b}{P}`는 두 점 A 와 B 의 아핀결합(affine combination) $P = (1 - b)A + bB$ 를 정의한다. 예를 들어, 그림 9의 코드 6째 줄과 동일한 표현들을 나열하면 다음과 같다.

```

\psLCNodeVar(A)(B)(1.2,-.2){Q}
\psLCNodeVar(A)(B)(!12 10 div 2 10 div neg){Q}
\psLCNode(A){1.2}{B}{-.2}{Q}
\psLNode(A)(B){-.2}{Q} %% affine combination
\nodexn{1.2(A)-.2(B)}{Q}

```

```

1 \begin{pspicture}[showgrid](4.5,4.5)
2 \psaxes[showorigin=false]{->}(4.5,4.5)
3 \pnode(1,3){A}\pnode(4,2){B}
4 \psline{o-o}(A)(B)\psdot(0)
5 \psLCNode(A){.5}(B){.5}{P}
6 \psLCNodeVar(A)(B)(1.2,-.2){Q}
7 \psLCNodeVar(A)(B) (!2 3 div 3 4 div){R}
8 %% some labeling codes skipped
9 \nodexn{.667(A)+0(B)}{M}\psdot(M)
10 \nodexn{0(A)+.75(B)}{N}\psdot(N)
11 \psline{->}(A)\psline{->}(B)
12 \psline[linestyle=dashed](M)(R)(N)
13 \end{pspicture}

```

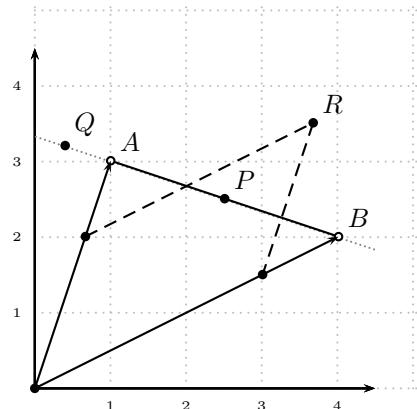


그림 9. 선형결합 노드

매크로 `\nodexn`은 `\nodexn{.5(A)-3(2,4)+.1(B)}{P}`와 같이 여러 점의 선형결합을 정의할 때도 사용한다.

4.2 벡터 연산을 이용한 노드

```
\AtoB(A)(B){C}
```

매크로 `\AtoB(A)(B){C}`는 두 벡터의 차 $B - A$ 를 노드좌표 C 로 정의한다. 예를 들어, $A = (1, 2)$, $B = (3, 1)$ 일 때 `\AtoB(A)(B){C}`는 $C = B - A = (2, -1)$ 을 정의한다. 이 예가 그림 10에 나타나 있다. 그림에서 코드 6째 줄의 `\AplusB`는 두 벡터의 합 $A + B$ 를 정의하며 코드 7째 줄의 `\midAB`는 중간점 $(A + B)/2$ 를 정의한다.

`\AtoB`가 정의하는 벡터는 위치뿐 아니라 이동도 의미하므로 임의의 점에서 다른 점으로의 방향을 쉽게 정의할 수 있어 유용하다. 그림 10에서 $\angle DAB$ 를 표시하기 위한 호는 어떻게 그렸을까? 먼저 코드 5째 줄과 12째 줄에서 매크로 `\AtoB`를 이용하여 노드좌표 (C)와 (F)로 방향을 정의한 후, 이를 13째 줄의 매크로 `\psarc`에서 노드좌표를 각도의 특수 표현으로 사용하여 그렸다.

4.3 나열과 함수를 이용한 노드

```
\pnodes{node root name}(x0,y0)(x1,y1)(x3,y3)...
```

예를 들어 설명하자면, `\pnodes{A}(1,2)(3;45)(2,2)...`는 노드좌표 (A0), (A1), (A2), ...를 나열된 좌표순서대로 정의한다. A는 좌표 이름의 뿌리이며 덧붙는 숫자는 0부터 차례로 증가한다. 그림 11의 코드 10–11째 줄에 그 활용 예가 나타나 있다.

```

\fnpnode{xval}{fn expr}{name}
\fnpnodes[plotpoints=num]{xmin}{xmax}{fn expr}{node root name}

```

```

1 \begin{pspicture}[showgrid](5,5)
2 \uput[1](0){$0$}
3 \dotnode(1,2){A}\dotnode(3,1){B}
4 \uput[u](A){$A$}\uput[d](B){$B$}
5 \AtoB(A)(B){C}
6 \AplusB(A)(B){D}
7 \midAB(A)(B){E}
8 \psdot(C)\uput[d](C){$C=B-A$}
9 \psdot(D)\uput[u](D){$D=A+B$}
10 \psdot(E)\uput[d](E){$E$}
11 %% \psline codes skipped
12 \AtoB(A)(D){F}
13 \psarc(A){10pt}{(C)}{(F)}
14 \uput[12pt][r](A){\tiny$\angle DAB$}
15 \end{pspicture}

```

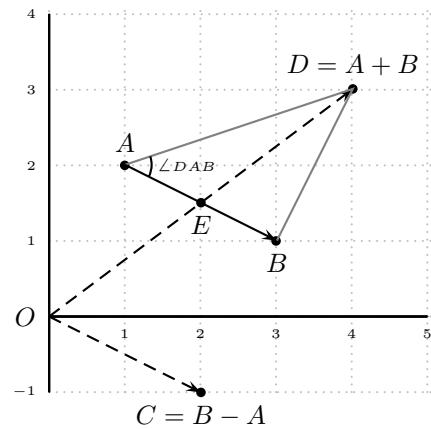


그림 10. 벡터 연산을 이용한 노드

매크로 `\fnpnode`는 함수 f 의 그래프상의 점 $(x, f(x))$ 를 정의한다. 함수 표현은 포스트스크립트 코드가 기본이며, 익숙한 대수적 표현을 쓰려면 `algebraic` 옵션을 주면 된다. `\fnpnode`의 활용 예를 그림 11의 코드 4-7째 줄에서 볼 수 있다. `\fnpnode`를 이용하여 좌표를 정의하는 코드 6째 줄을 `algebraic` 옵션을 사용하지 않고 포스트스크립트 코드로 쓰면 다음과 같다.

```
\fnpnode[algebraic=false]{3.5}{x dup 1 add mul 5 div}{A}
```

그래프 상의 여러 개의 점을 표시하려면 `\fnpnodes`를 이용하면 편리하다. 파라미터 `plotpoints`의 값인 `num`은 `xmin`에서 `xmax`까지의 범위에서 몇 개의 점을 지정할 것인가를 말해준다. `\fnpnodes`의 경우에도 `algebraic` 옵션을 주면 대수적 함수 표현을 쓸 수 있다. 그림 11의 코드 8-9째 줄은 그래프상에 P0에서 P4까지 5개의 점을 지정하고 찍는 것을 보여주고 있다.⁹

```

1 \begin{pspicture}[showgrid](4,4)
2 \psaxes{->}(4.5,4.5)
3 \def\Fxi{x*(x+1)/5}
4 \psset{algebraic}
5 \psplot[linewidth=1.2pt]{0}{4}{\Fxi}
6 \fnpnode{3.5}{\Fxi}{A}
7 \psdot{dotstyle=o}(A)\uput[u](A){$A$}
8 \fnpnodes[plotpoints=5]{1}{3}{\Fxi}{P}
9 \multido{\n=0+1}{5}{\psdots(P\n)}
10 \pnodes{B}{(2,4)(0,.5)(4,1.5)(3;90)(2.5,0)}
11 \pspolygon{showpoints}(B0)(B1)(B2)(B3)(B4)
12 \end{pspicture}

```

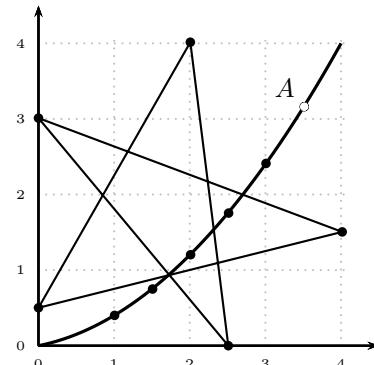


그림 11. 나열과 함수를 이용한 노드

9. `\multido`에 관해서는 [8]을 보라.

4.4 상대 위치 노드

```
\psRelNode [angle=value] (A)(B){length factor}{endnodeName}
\psRelNodeVar (A)(B)(radius;angle){name}
```

매크로 `\psRelNode`와 `\psRelNodeVar`는 임의의 점을 중심으로 상대적인 위치를 정하고 이름을 붙인다.¹⁰ 구체적으로 이 매크로는 점 A를 중심으로, 선분 AB의 연장선을 기준선으로, 그리고 선분 AB의 길이를 단위로 하여 `length factor` 또는 `radius`만큼의 길이를 `angle`만큼 회전시킨 점을 정의한다. 그림 12에서 \overline{AP} 는 \overline{AB} 의 길이의 2배이며 $\angle PAQ = 30^\circ$ 이고 $\angle PAR = 90^\circ$ 이다.

두 매크로는 같은 기능을 하지만 `\psRelNodeVar`의 경우 그림 12의 코드 6째 줄에서 보듯이 `(radius;angle)`의 표현에 `\SpecialCoor`가 허용하는 특수 표현을 사용할 수 있다.

```
1 \begin{pspicture}[showgrid](4,4)
2 \psaxes(4,4)
3 \pnode(1,2){A}\pnode(2,1){B}
4 \psRelNode [angle=0] (A)(B){2}{P}
5 \psRelNode [angle=30] (A)(B){2}{Q}
6 \psRelNodeVar (A)(B)(2;!30 60 add){R}
7 \psline(A)(B)
8 \psline(A)(Q)
9 \psdots(A)(B)(P)(Q)(R)
10 \psline[linestyle=dotted](A)(P)(R)(A)
11 %% labeling codes skipped
12 \end{pspicture}
```

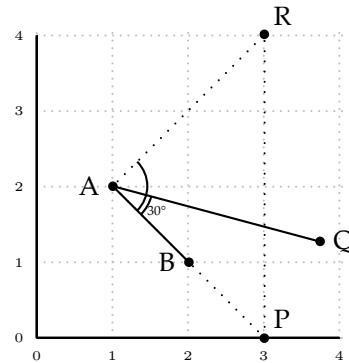


그림 12. 상대 위치 노드

4.5 교차점 노드

```
\psIntersectionPoint(P0)(P1)(P2)(P3){name}
```

매크로 `\psIntersectionPoint`는 두 직선의 교차점을 노드좌표로 정의한다.¹¹ 위 구문에서 처음 두 좌표 P0과 P1은 한 직선상의 점이며 나머지 두 좌표 P2와 P3는 다른 직선상의 점이다. 그림 13은 두 점을 연결하는 선과 또 다른 두 점을 연결하는 선의 교차점을 노드좌표 E로 정의하는 예를 보여주고 있다.

4.6 노드의 성분 분리

```
\psGetNodeCenter{nameNode}
```

10. 매크로 `\psRelNode`에 대해서는 pstricks-add 패키지 매뉴얼 [4]를 보라.

11. 매크로 `\psIntersectionPoint`에 대해서는 pstricks-add 패키지 매뉴얼 [4, p.29]를 보라.

```

1 \begin{pspicture}[showgrid](4,4)
2 \psaxes(4,4)
3 \pnodes{P}(1,3)(4,1)(1,1)(2,3)
4 \psline(P0)(P1)\psline(P2)(P3)
5 \psdots(P0)(P1)(P2)(P3)
6 \psIntersectionPoint(P0)(P1)(P2)(P3){E}
7 \psdot(E)\put[-80]{(E)}{E}
8 \put[200]{(P0)}{P0}\put[200]{(P1)}{P1}
9 \put[1]{(P2)}{P2}\put[u]{(P3)}{P3}
10 \end{pspicture}

```

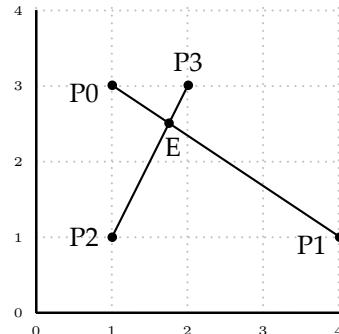


그림 13. 교차점 노드

매크로 `\psGetNodeCenter{A}`는 내부적으로 노드 A의 각 성분인 `A.x`와 `A.y`를 정의한다. 사용자는 이 정의된 값을 사용할 수 있다. 다만, 포스트스크립트 수준에서만 작동한다.

그림 14의 코드 4-5째 줄은 노드 A에 대해 상대적으로 $(3, -3)$ 만큼 이동하는 점을 새로운 노드로 정의하는 데에 `\psGetNodeCenter`를 사용하고 있다.

```

1 \begin{pspicture}[showgrid](4,4)
2 \psaxes(4,4)
3 \pnode(0.5,3.5){A}\psdot(A)
4 \pnode{! \psGetNodeCenter{A}
      A.x 3 add A.y 3 sub}{RelA}
5 \psdot(RelA)
6 \ncline[nodesepB=1pt]{>}{A}{RelA}
7 \put[30]{(A)}{A}\put[30]{(RelA)}{RelA}
8 \end{pspicture}

```

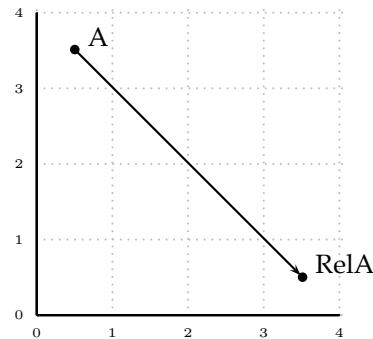


그림 14. 노드의 성분 분리

5 함수의 플로팅 활용

5.1 일차연립방정식의 그래프 해법

여기에서는 두 개의 일차식으로 이루어진 연립방정식의 해를 그래프 상에 나타내는 방법을 살펴보기로 한다. 다음의 일차연립방정식을 예로 들자.

$$4x + 3y = 12 \quad (1)$$

$$2x - 2y = -1 \quad (2)$$

x 를 거래량, y 를 가격으로 해석하면 식 (1)은 시장수요(market demand), 식 (2)는 시장공급(market supply)을 나타낸다고 할 수 있다.

이를 그림으로 나타내는 그림 15를 보자. 식 (1)을 $y = f(x)$, 식 (2)를 $y = g(x)$ 의 형식으로 바꿔 코드 4째 줄에 정의해 두고, 이어 코드 5째 줄에서 각 함수의 그래프를 그린다.¹² 코드 6~7째 줄은 매크로 \pnodes를 이용하여 각 그래프상에서 두 점씩을 뽑아 노드좌표를 정의한다. 노드좌표의 정의에는 $(*x \{f(x)\})$ 형식의 대수적 포스트스크립트 노드를 썼다. 코드 9째 줄은 이 점들로부터 매크로 \psIntersectionPoint를 이용하여 해 (x^*, y^*) 를 구하고 이를 E 라 이름 붙인다. 해(E)와 원점(O)으로 혼합좌표를 구성하면 좌표 $(x^*, 0)$ 은 $(E | O)$ 로, 좌표 $(0, y^*)$ 는 $(O | E)$ 로 정의하여 사용할 수 있다. 코드 11~12째 줄이 그 활용 예이다.

```

1 \begin{pspicture}[showgrid] (4,4)\scriptsize
2 \psset{algebraic}
3 \psaxes{->}(4,4.5)[$x$,0][$y$,90]
4 \def\fxi{4-4/3*x}\def\gxi{x+.5}
5 \psplot{0}{3}{\fxi}\psplot{0}{3.5}{\gxi}
6 \pnodes{F}{*0 {\fxi}}{*1 {\fxi}}
7 \pnodes{G}{*0 {\gxi}}{*1 {\gxi}}
8 \psdots(F0)(F1)(G0)(G1)
9 \psIntersectionPoint(F0)(F1)(G0)(G1){E}
10 \psdot(E)\put[r](E){$E$}
11 \psline[linestyle=dashed](0|E)(E)(E|0)
12 \put[d](E|0){$x^*$}\put[d1](0|E){$y^*$}
13 \put[u1](F0){F0}\put[1](G0){G0}
14 \put[80](F1){F1}\put[-80](G1){G1}
15 \end{pspicture}
```

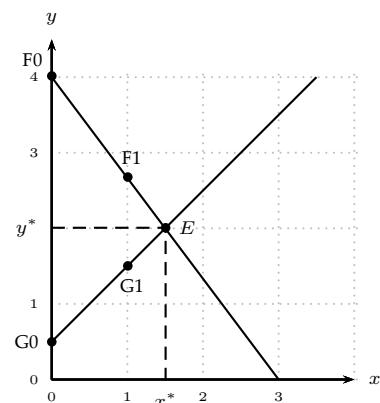


그림 15. 일차연립방정식의 해

식 (1)과 (2)를 수요공급모형으로 해석하면 y^* 는 시장균형가격, x^* 는 시장균형거래량을 의미한다.

절편 나타내기와 몇 가지 팁

수요공급모형을 그릴 때 그래프의 절편(intercept)을 표시해야 할 때가 있다. 또한 최저가격제나 최고가격제 등을 그림으로 나타낼 때에는 일정한 함수값에 대응하는 그래프상의 점 등을 표시할 필요가 있다. 이와 관련한 몇 가지 팁을 논의하고자 한다.

먼저 그림 15의 함수에서 각 축에 대한 절편을 나타내는 문제를 그림 16을 통해 살펴보자. 그림 16의 코드에는 논의하고자 하는 코드만 남기고 다른 코드는 생략하였다. y -절편은 함수 값 $f(0)$ 와 $g(0)$ 이므로 이에 대응하는 노드는 각각 그림 16의 코드 4~5째 줄에서 정의하는 F0 과 G0이다. 코드 6째 줄은 단순히 매크로 \psIntersectionPoint의 이름을 \psIP로 줄이기 위한 것이다.

x -절편은 x -축의 두 점과 각 그래프상의 두 점의 교차점으로 나타내면 된다. 그림 16의 코드 7~8째 줄은 각 절편에 상응하는 좌표를 Fx와 Gx로 정의한다.

12. 식 (1)과 (2)를 수요공급모형으로 해석할 때, 엄밀하게 말하면, $y = f(x)$ 와 $y = g(x)$ 는 각각 역수요함수(inverse demand function)와 역공급함수(inverse supply function)이다.

임의의 x 값에 대응하는 그래프상의 점은 함수값 $f(x), g(x)$ 를 이용하여 쉽게 나타낼 수 있다. 임의의 함수값, 예를 들어 $y = 1$ 에 대응하는 그래프상의 점은 어떻게 나타낼까? 이는 $y = 1$ 을 나타내는 선 위의 두 점, 예를 들어 $(0, 1)$ 과 $(1, 1)$ 을 잡아 \psIntersectionPoint를 이용하여 나타낼 수 있다. 그림 16의 코드 9째 줄은 $y = 1$ 에 대응하는 함수 g 의 그래프상의 점을 정의하고 A 로 이름 붙인다.

```

1  %% codes in issue are only shown
2  \def\Fxi{4-4/3*x}\psplot{-5}{3}{\Fxi}
3  \def\Gxi{x+.5}\psplot{-5}{3.5}{\Gxi}
4  \pnodes{F}{*0 {\Fxi}}(*1 {\Fxi})
5  \pnodes{G}{*0 {\Gxi}}(*1 {\Gxi})
6  \def\psIP{\psIntersectionPoint} % short
7  \psIP(0,0)(1,0)(F0)(F1){Fx}
8  \psIP(0,0)(1,0)(G0)(G1){Gx}
9  \psIP(0,1)(1,1)(G0)(G1){A} % y fixed at 1
10 \psIP(A)(0|A)(F0)(F1){B}
11 \psIP(B)(B|0)(G0)(G1){C}
12 \psset{linestyle=dotted,dotsep=1pt}
13 \psline(-.2,0|A)(3,0|A)
14 \psline(B|0,-.2)(B|0,4)

```

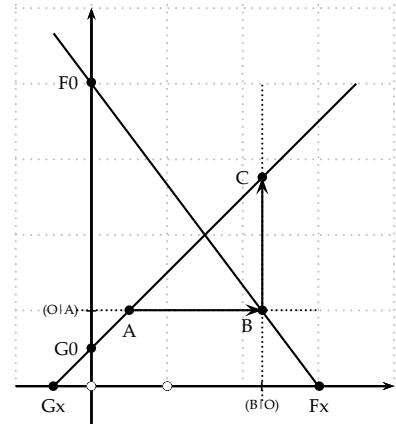


그림 16. 절편 나타내기와 몇 가지 텁

이제 임의의 점, 예를 들어 A 가 주어졌다고 하자. 이 점으로부터 축과 평행하게 움직일 때 다른 그래프와 만나는 점은 어떻게 나타낼까? 이는 혼합좌표를 이용하면 쉽게 정의할 수 있다. 그림 16의 코드 10째 줄은 A 로부터 수평방향으로 움직일 때 함수 f 의 그래프와 만나는 점 B 를 정의하고 있으며, 11째 줄은 B 로부터 수직방향으로 움직여 함수 g 의 그래프와 만나는 점 C 를 정의하고 있다.

매크로 \psIntersectionPoint를 이용하여 그래프상의 점을 찾는 것은 편리하지만 이는 직선의 경우에만 적용할 수 있다. 만일 그림 16의 점 B 의 x -성분만 뽑아낼 수 있으면 C 를 $(x, g(x))$ 와 같이 정의하는 것이 보다 일반적인 방법이다. 매크로 \psGetNodeCenter를 이용하면 이와 같은 방법을 사용할 수 있다. 단지 포스트스크립트 수준에서 작동한다는 것을 잊지 않아야 한다. 그림 16의 코드 11째 줄을 다음으로 대체해도 같은 결과를 얻는다.

```

\def\Fx{4 dup 3 div x mul sub}          % f(x)
\pnode(!\psGetNodeCenter{B} B.x \Fx){C} % C=(x,f(x))

```

마지막으로, 임의의 점 A 를 지나는 선을 긋는 방법을 살펴보자. 그림 7에서 보는 특수노드좌표 ([par]A)나 그림 12에서 보는 \psRelNode 등을 이용하면 원하는 방향으로 원하는 길이의 선을 그릴 수 있다. 하지만 축과 평행한 선을 긋는다면 간단히 혼합좌표를 이용할 수 있다. 예를 들어 점 A 를 지나고 x 축에 평행한 선을 $x = a$ 에서 $x = b$ 까지 그린다면 \psline(a,0|A)(b,0|A)와 같이 하면 된다. 그림 16의 코드 12-14째 줄이 그 활용 예이다.

5.2 몇 가지 경제학 모형

독점 모형

여기에서는 독점 모형과 균형상황을 그림으로 나타내는 문제를 살펴보자 한다.

비용함수가 $C(Q)$ 인 기업이 직면하는 시장수요를 역수요함수 $P = P(Q)$ 로 나타내자. 이 독점기업의 수입은 $R(Q) = P(Q)Q$ 로 나타낼 수 있다. 이 독점기업의 문제는 이윤 $\pi(Q) = R(Q) - C(Q)$ 를 극대화하는 생산량 Q^* 를 선택하는 것이다. 이 기업의 이윤극대문제는 다음과 같다.

$$\max_Q \pi(Q) = R(Q) - C(Q)$$

이 기업의 이윤극대조건은 $\pi'(Q) = R'(Q) - C'(Q) = 0$ 이며, 한계수입(marginal revenue)을 $MR(Q) \equiv R'(Q)$, 한계비용(marginal cost)을 $MC(Q) \equiv C'(Q)$ 와 같이 바꿔 쓰면, 이윤극대생산량은 $MR(Q) = MC(Q)$ 를 만족해야 한다.

구체적으로 시장수요를 $P = 70 - Q$, 비용함수를 $C(Q) = Q^2 - 5Q + 200$ 이라 하자. 그러면 한계수입은 $MR(Q) = 70 - 2Q$, 한계비용은 $MC(Q) = 2Q - 5$ 이며 평균비용(average cost)은 $AC(Q) = Q - 5 + 200/Q$ 이다.

그림 17에 이 모든 함수의 그래프가 그려져 있다. 코드 2~5째 줄은 수요함수, 한계수입, 한계비용, 평균비용함수를 대수적 방식으로 정의한다. 코드 11째 줄은 균형생산량(Q^*)을 찾기 위한 균형점 E 를 구하고, 코드 12째 줄은 균형가격(P^*)을 찾기 위한 노드 A 를 정의한다. 코드 13째 줄은 평균비용함수를 포스트스크립트 코드로 다시 정의하고 있다. 이는 균형(E)에서 x -좌표를 추출하여 평균비용의 값 $AC(Q^*)$ 를 구하기 위해 `\psGetNodeCenter`를 이용할 때 필요하기 때문이다(코드 13~15째 줄 참고). 네 점 (A) ($A|0$) (0) ($0|A$)로 이루어지는 사각형의 면적은 수입의 크기이며 이윤은 사각형 (A) (B) ($0|B$) ($0|A$)로 나타낼 수 있다.

```

1  %% main codes are only shown
2  \def\DM{70-x}
3  \def\MR{70-2*x}
4  \def\MC{2*x-5}
5  \def\AC{x-5+200/x}
6  %% \psplot codes skipped
7  \pnodes{D}{*0 {\DM}}(*1 {\DM})
8  \pnodes{MR}{*0 {\MR}}(*1 {\MR})
9  \pnodes{MC}{*0 {\MC}}(*1 {\MC})
10 \def\psIP{\psIntersectionPoint}
11 \psIP(MR0)(MR1)(MC0)(MC1){E}
12 \psIP(E)(E|0)(D0)(D1){A}
13 \def\ACx{/x exch def x 5 sub 200 x div add }
14 \def\psGNC{\psGetNodeCenter}
15 \pnode(!\psGNC{E} E.x dup \ACx){B}
```

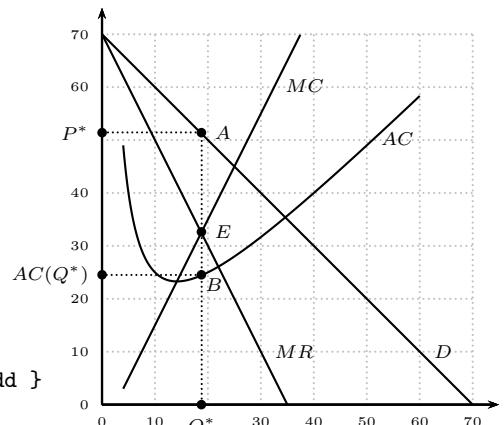


그림 17. 독점 모형

쿠르노(Cournot) 복점 모형

기업 1과 2의 생산량이 각각 q_1 과 q_2 일 때 두 기업의 비용을 각각 $c_1(q_1) = 4q_1$ 과 $c_2(q_2) = 8q_2$ 이라 하자. 두 기업이 조업하는 복점시장의 수요를 $P = 20 - Q$ 로 나타낼 수 있다고 하자 ($Q = q_1 + q_2$). 두 기업은 상대의 의사결정을 모르는 상태에서 자신의 이윤을 가장 크게 하는 생산량을 결정하고자 한다. 기업 1의 이윤극대화문제는 주어진 q_2 에 대해 다음과 같다.

$$\max_{q_1} \pi_1(q_1, q_2) = (20 - (q_1 + q_2))q_1 - 4q_1$$

마찬가지로 기업 2의 문제도 나타낼 수 있다.

각 극대화문제의 일차조건 $\frac{\partial \pi_i}{\partial q_i} = 0$ 으로부터 각 기업 i 의 최적반응함수(best response function)를 구하면 다음과 같다.

$$q_1 = BR^1(q_2) = \frac{16 - q_2}{2} \quad (3)$$

$$q_2 = BR^2(q_1) = \frac{12 - q_1}{2} \quad (4)$$

식 (3)과 (4)를 연립으로 풀면 내쉬균형(Nash Equilibrium: NE) (q_1^*, q_2^*) 를 구할 수 있다.

```

1 \begin{pspicture}(13,17)\scriptsize
2 \psset{algebraic}
3 \psaxes[Dx=2,Dy=2]{->}(13,17)[\$q_1\$,0][\$q_2\$,90]
4 \def\Fx{16-2*x} % BR1(inverse)
5 \def\Gx{(12-x)/2} % BR2
6 \psplot{0}{8}{\Fx}
7 \psplot{0}{12}{\Gx}
8 \pnodes{F}{*0 {\Fx}}{*1 {\Fx}}
9 \pnodes{G}{*0 {\Gx}}{*1 {\Gx}}
10 \psIntersectionPoint(F0)(F1)(G0)(G1){NE}
11 %% labeling codes skipped
12 \end{pspicture}
```

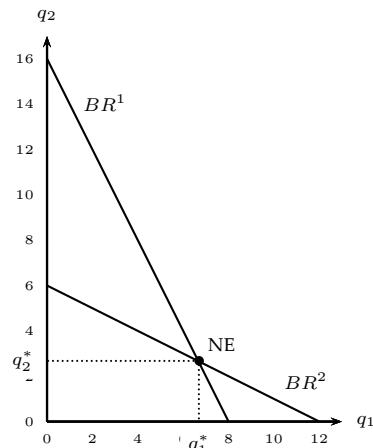


그림 18. 쿠르노 복점 모형 : 내쉬균형

그림 18에 최적반응함수와 내쉬균형이 그려져 있다. 그림에는 q_1 이 가로축에, q_2 가 세로축에 나타나 있다. 식 (3)의 그래프는 다음과 같이 swapaxes 옵션을 사용하여 그릴 수 있다.

```
\psplot[algebraic,swapaxes]{0}{16}{(16-x)/2}
```

그러나 \psIntersectionPoint를 이용하여 노드를 정의하는 등 함수의 정보를 이용하려면 코드 4째 줄에서 보는 바와 같이 역함수 $q_2 = 16 - 2q_1$ 로 바꿔 정의해두는 것이 편리하다.

그림 19는 식 (3)과 (4)가 이루어내는 동태적 과정을 나타내고 있다. 그림에서 점선으로 표시된 보조선을 지우려면 코드 18–19째 줄만 지우면 된다. 그림 19를 플로팅할 때에는 복잡한 과정을 그려내기 위해 포스트스크립트 언어를 그대로 썼다. 이 코드의 5–7째 줄의 함수만

바꾸면 유사한 결과를 얻을 수 있다. 물론, 이 코드를 사용할 때에는 1-3째 줄의 그래픽 크기나 범위와 10-11째 줄의 플로팅 범위를 상황에 맞게 적절하게 조절해야 할 것이다.

```

1  \psset{unit=.9,arrowscale=1.5}
2  \begin{pspicture}(-1,-1)(13,17)
3  \psaxes[Dx=2,Dy=2]{->}(0,0)(-1,-1)(13,17)[\$q_1$,-45][\$q_2$,135]
4  % defining functions
5  \def\Fx{\ /xVal\ exch\ def\ 16\ 2\ xVal\ mul\ sub\ }% BR1(inverse)
6  \def\FxINV{\ /yi\ exch\ def\ 8\ 0.5\ yi\ mul\ sub\ }% BR1
7  \def\Gx{\ /xx\ exch\ def\ 6\ 0.5\ xx\ mul\ sub\ }% BR2
8  \def\GxVal{\ 6\ 0.5\ x\ mul\ sub\ }% BR2
9  % plotting
10 \psplot[linewidth=1.5pt]{0}{12}{x \Gx}
11 \psplot[linewidth=1.5pt]{0}{8}{x \Fx}
12 % dynamics starting
13 \pstVerb{ /x 1 def }% initial value
14 \psline[linewidth=1.2pt]{->}(! 0 x \Fx)(! x dup \Fx)
15 \psline[linewidth=.4pt]{->}(! x dup \Fx)(! x \GxVal)
16 % dynamics repeated
17 \multido{\n=0+1}{5}{%
18   \psline[linestyle=dotted](! x 0)(! x \GxVal)
19   \psline[linestyle=dotted](! 0 \GxVal)(! x \GxVal)
20   \psline[linewidth=1.2pt](! x \GxVal)(! \GxVal \FxINV \GxVal)
21   \psline[linewidth=.4pt](! \GxVal \FxINV \GxVal)(! \GxVal \FxINV dup \Gx)
22   \pstVerb{ /x \GxVal \FxINV def }
23 }
24 \end{pspicture}
```

6 맷음말

PSTricks 패키지는 데카르트 좌표 체계와 극좌표 체계를 사용한다. 극좌표를 사용하려면 `\SpecialCoor`를 선언해야 하지만, 함수를 플로팅하기 위한 대부분의 경우 이 명령이 기본으로 설정된 패키지를 사용하므로, 크게 신경 쓸 일은 아니다. `\SpecialCoor`는 데카르트 좌표나 극좌표를 표현함에 있어서 통상적으로 익숙한 좌표의 표현 이외에 매우 다양한 특수 표현을 사용할 수 있게 해준다.

그림을 그릴 때 간단하게는 모눈종이를 이용하여 좌표를 일일이 찍는 것도 한 방법이지만, 이 경우는 그림에 간단한 수정을 하거나 유사한 그림을 그릴 때에 다시 사용할 수 없어 효율적이라 할 수 없다. `\SpecialCoor`가 허용하는 특수좌표를 이해하고 익히는 것은 PSTricks 패키지의 플로팅 기능을 충분히 사용하고 함수의 플로팅 작업을 정확하고 효율적으로 하기 위해 필요한 일이다.

이 글에서 예로 든 코드들은 그림의 크기나 플로팅의 범위들을 조절한다면 유사한 그림을 플로팅하기 위해 재사용할 수 있다. 이 글이 함수 플로팅을 효율적으로 하고자 하는 독자에게, 적어도 경제학 분야에서 함수를 플로팅하고자 하는 독자에게 도움이 되기를 기대한다.

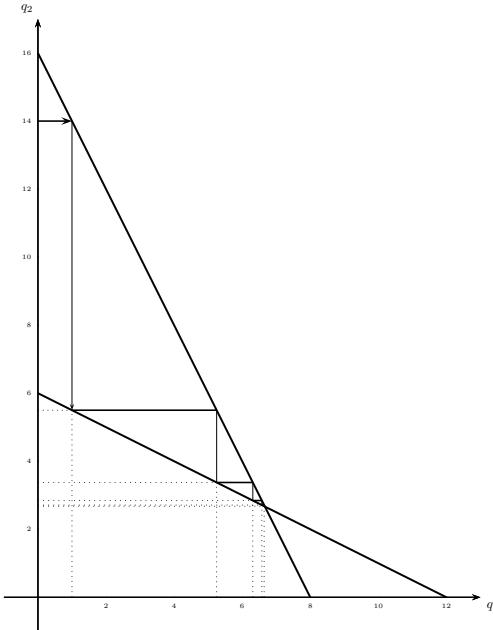


그림 19. Cournot 복점 모형 : dynamics

참고 문헌

1. 조인성, PSTricks를 이용한 함수의 플로팅 : 효용극대화 모형을 중심으로, *The Asian Journal of TeX* 1 (2007), no. 1, 85–109.
2. Bill Casselman, *Mathematical Illustrations: A Manual of Geometry and PostScript*, Cambridge Univ. Press, 2005.
3. Michel Goossens, Frank Mittelbach, Sebastian Rahtz, Denis Roegel, and Herbert Voß, *The LATEX Graphics Companion*, 2nd ed., Addison-Wesley, 2007.
4. Dominique Rodriguez, Michael Sharpe, and Herbert Voß, *pstricks-add: addtionals Macros for pstricks*, ver. 3.46, December 13, 2010. CTAN:graphics/pstricks/contrib/pstricks-add/pstricks-add-doc.pdf
5. Herbert Voß, *3D plots: pst-3dplot*, ver. 1.90, February 18, 2010. CTAN:graphics/pstricks/contrib/pst-3dplot/pst-3dplot-doc.pdf
6. _____, *pst-func : plotting special mathematical functions*, ver. 0.71, November 3, 2010. CTAN:graphics/pstricks/contrib/pst-func/
7. Timothy Van Zandt, *PSTricks: PostScript macros for Generic TeX. User's Guide*, ver. 97, July 25, 2003. CTAN:graphics/pstricks/base/doc/pstricks-doc.pdf
8. _____, *Documentation for multido.tex: A loop macro for Generic TeX*, ver. 1.42, May 14, 2010. CTAN:macros/generic/multido/multido-doc.pdf
9. Timothy Van Zandt and Herbert Voß, *pst-plot: plotting data and functions*, ver. 1.25, December 14, 2010. CTAN:graphics/pstricks/contrib/pst-plot/pst-plot-doc.pdf
10. Timothy Van Zandt, Michael Sharpe, and Herbert Voß, *pst-node: Nodes and node connections*, ver. 1.14, December 28, 2010. CTAN:graphics/pstricks/contrib/pst-node/pst-node-doc.pdf