# Two applications of macros in PSTricks*
# &
# How to color arrows properly for a vector field

Le Phuong Quan
(Cantho University, Vietnam)
lpquan@ctu.edu.vn

October 3, 2013

## Contents

**1 Drawing approximations to the area under a graph by rectangles** **1**
- 1.1 Description . . . . . 1
- 1.2 Examples . . . . . 4

**2 Drawing the vector field of an ordinary differential equation of order one** **5**
- 2.1 Description . . . . . 5
- 2.2 Examples . . . . . 7

**3 Remarks on how to color arrows properly for a vector field** **10**
- 3.1 Description . . . . . 10
- 3.2 Examples . . . . . 12

**Acknowledgment** **13**

## 1 Drawing approximations to the area under a graph by rectangles

### 1.1 Description

We recall here an application in Calculus. Let $f(x)$ be a function, defined and bounded on the interval $[a, b]$. If $f$ is integrable (in Riemann sense) on $[a, b]$, then its definite integral over this interval is

$$\int_a^b f(x)dx = \lim_{\|P\|\to 0} \sum_{i=1}^{n} f(\xi_i)\Delta x_i,$$

*PSTricks is the original work of Timothy Van Zandt (email address: `tvz@econ.insead.fr`). It is currently edited by Herbert Voß (`hvoss@tug.org`).

where $P\colon a = x_0 < x_1 < \cdots < x_n = b$, $\Delta x_i = x_i - x_{i-1}$, $\xi_i \in [x_{i-1}, x_i]$, $i = 1, 2, \ldots, n$, and $\|P\| = \max\{\Delta x_i\colon i = 1, 2, \ldots, n\}$. Hence, when $\|P\|$ is small enough, we may have an approximation

$$I = \int_a^b f(x)dx \approx \sum_{i=1}^n f(\xi_i)\Delta x_i. \tag{1}$$

Because $I$ is independent to the choice of the partition $P$ and of the $\xi_i$, we may divide $[a, b]$ into $n$ subintervals with equal length and choose $\xi_i = (x_i + x_{i-1})/2$. Then, $I$ can be approximately seen as the sum of areas of the rectangles with sides $f(\xi_i)$ and $\Delta x_i$.

We will make a drawing procedure to illustrate the approximation (1). Firstly, we establish commands to draw the *sum* of rectangles, like the area under piecewise-constant functions (called *step shape*, for brevity). The choice here is a combination of the macros `\pscustom` (to *join* horizontal segments, automatically) and `\multido`, of course. In particular, the horizontal segments are depicted within the loop `\multido` by

`\psplot[`*settings*`]`$\{x_{i-1}\}\{x_i\}\{f(\xi_i)\}$

The `\pscustom` will join these segments altogether with the end points $(a, 0)$ and $(b, 0)$, to make the boundary of the step shape. Then, we draw the points $(\xi_i, f(\xi_i))$, $i = 1, 2, \ldots, n$, and the dotted segments between these points and the points $(\xi_i, 0)$, $i = 1, 2, \ldots, n$, by

`\psdot[algebraic,...](*`$\{\xi_i\}$ $\{f(x)\}$`)`,
`\psline[algebraic,linestyle=dotted,...](`$\xi_i$`,0)(*`$\{\xi_i\}$ $\{f(x)\}$`)`,

where we use the structure `(*`$\{$*value*$\}$ $\{f(x)\}$`)` to obtain the point $(\xi_i, f(\xi_i))$. Finally, we draw vertical segments to split the step shape into rectangular cells by

`\psline[algebraic,...](`$x_i$`,0)(*`$\{x_i\}$ $\{f(x - \Delta x_i/2)\}$`)`

The process of performing steps is depicted in Figure 1.

We can combine the above steps to make a procedure whose calling sequence consists of main parameters $a$, $b$, $f$ and $n$, and dependent parameters $x_{i-1}$, $x_i$, $\xi_i$, $f(\xi_i)$ and $f(x \pm \Delta x_i/2)$. For instant, let us consider the approximations to the integral of $f(x) = \sin x - \cos x$ over $[-2, 3]$ in the cases of $n = 5$ and $n = 20$. Those approximations are given in Figure 2.

In fact, we can make a procedure to illustrate the approximation (1), say `RiemannSum`, whose calling sequence has the form

`\RiemannSum`$\{a\}\{b\}\{f(x)\}\{n\}\{x_{\text{ini}}\}\{x_{\text{end}}\}\{x_{\text{choice}}\}\{f(x+\Delta x_i/2)\}\{f(x-\Delta x_i/2)\}$,

where $x_0 = a$ and for each $i = 1, 2 \ldots, n$:

$$x_i = a + \frac{b-a}{n}i, \quad \Delta x_i = x_i - x_{i-1} = \frac{b-a}{n},$$
$$x_{\text{ini}} = x_0 + \Delta x_i, \quad x_{\text{end}} = x_1 + \Delta x_i, \quad x_{\text{choice}} = \frac{x_{\text{ini}} + x_{\text{end}}}{2} = \frac{x_0 + x_1}{2} + \Delta x_i.$$

Note that $x_{\text{ini}}$, $x_{\text{end}}$ and $x_{\text{choice}}$ are given in such forms to be suitable to variable declaration in `\multido`. They are nothing but $x_{i-1}$, $x_i$ and $\xi_i$, respectively, at the step $i$-th in the loop.

Tentatively, in `PSTricks` language, the definition of `RiemannSum` is suggested to be

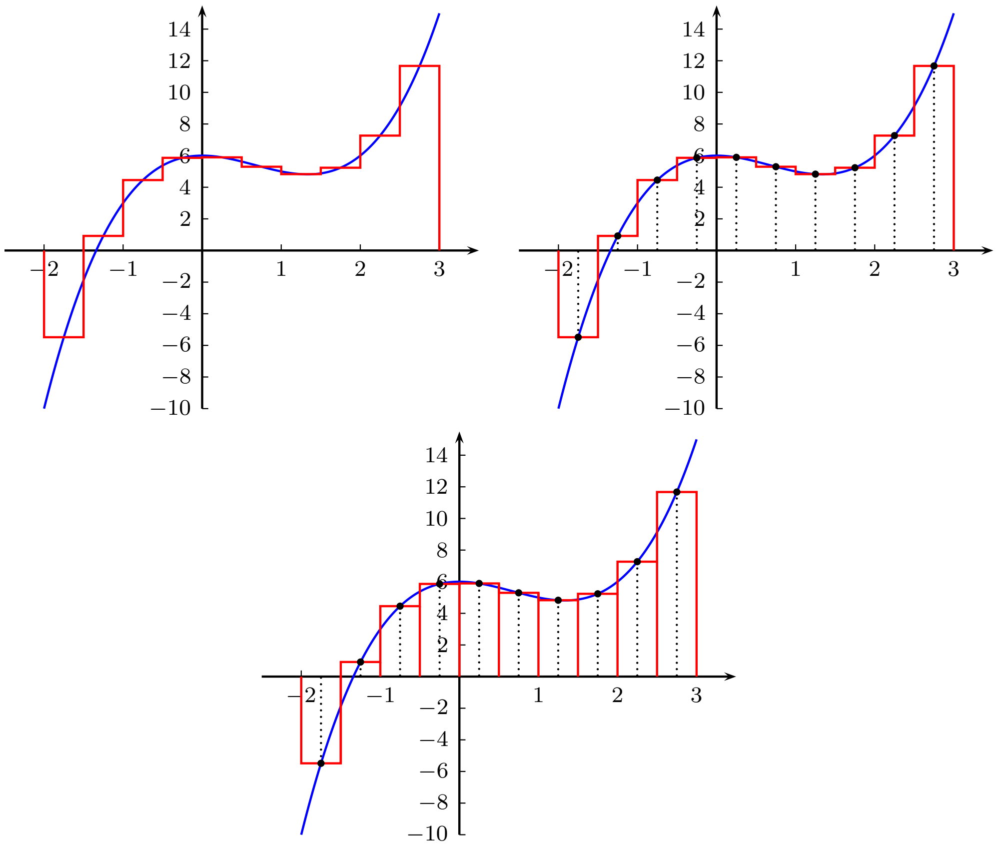


Figure 1: Steps to make the drawing procedure.

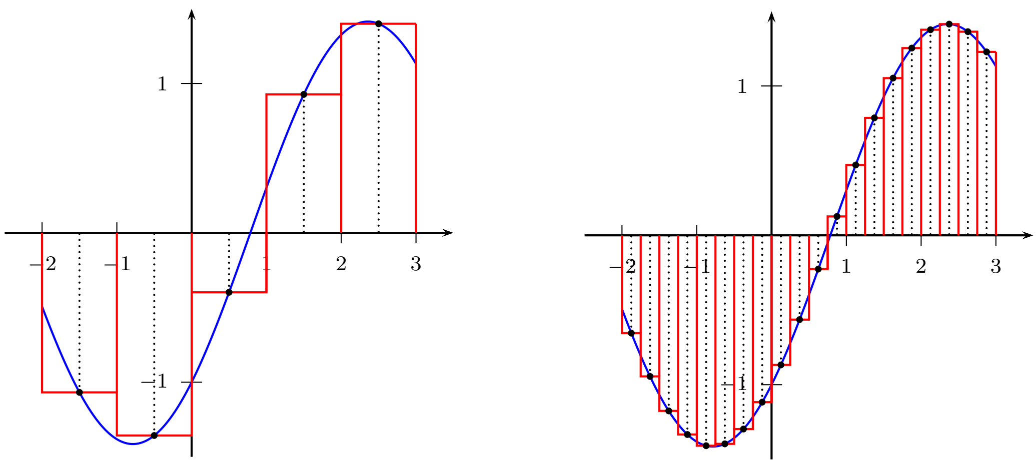


Figure 2: Approximations to the integral of $f(x) = \sin x - \cos x$ over $[-2, 3]$.

```
\def\RiemannSum#1#2#3#4#5#6#7#8#9{%
\psplot[linecolor=blue]{#1}{#2}{#3}
\pscustom[linecolor=red]{%
\psline{-}(#1,0)(#1,0)
\multido{\ni=#5,\ne=#6}{#4}
{\psline(*{\ni} {#8})(*{\ne} {#9})}}
\multido{\ne=#6,\nc=#7}{#4}
{\psdot(*{\nc} {#3})
\psline[linestyle=dotted,dotsep=1.5pt](\nc,0)(*{\nc} {#3})
\psline[linecolor=red](\ne,0)(*{\ne} {#9})}}
```

### 1.2 Examples

We give here two more examples just to see that using the drawing procedure is very easy. In the first example, we approximate the area under the graph of the function $f(x) = x - (x/2)\cos x + 2$ on the interval $[0, 8]$. To draw the approximation, we try the case $n = 16$; thus $x_0 = 0$ and for each $i = 1, \ldots, 16$, we have $x_i = 0.5\,i$, $\Delta x_i = 0.5$, $x_{\text{ini}} = 0.00 + 0.50$, $x_{\text{end}} = 0.50 + 0.50$ and $x_{\text{choice}} = 0.25 + 0.50$.

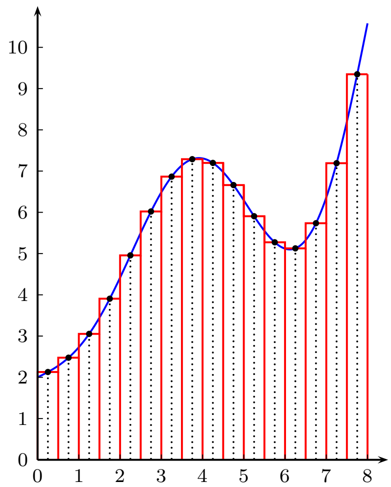


Figure 3: An approximation to the area under the graph of $f(x) = x - (x/2)\cos x + 2$ on $[0, 8]$.

To get Figure 3, we have used the following LaTeX code:

```
\begin{pspicture}(0,0)(4.125,5.5)
\psset{plotpoints=500,algebraic,dotsize=2.5pt,unit=0.5}
\RiemannSum{0}{8}{x-(x/2)*cos(x)+2}{16}{0.00+0.50}{0.50+0.50}{0.25+0.50}
{x+0.25-((x+0.25)/2)*cos(x+0.25)+2}{x-0.25-((x-0.25)/2)*cos(x-0.25)+2}
\psaxes[ticksize=2.2pt,labelsep=4pt]{->}(0,0)(8.5,11)
\end{pspicture}
```

In the second example below, we will draw an approximation to the integral of $f(x) = x \sin x$ over $[1, 9]$. Choosing $n = 10$ and computing parameters needed, we get Figure 4, mainly by the command

$$\texttt{\textbackslash RiemannSum}\{1\}\{9\}\{x\sin x\}\{10\}\{1.00+0.80\}\{1.80+0.80\}\{1.40+0.80\}$$
$$\{(x+0.4)\sin(x+0.4)\}\{(x-0.4)\sin(x-0.4)\}$$

in the drawing procedure.

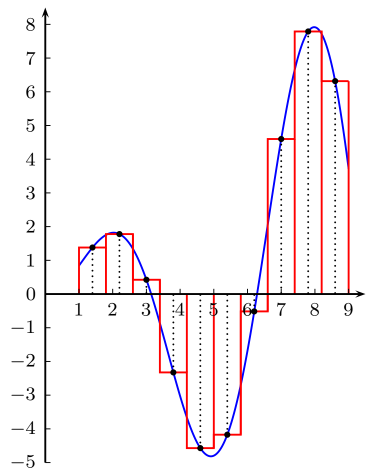


Figure 4: An approximation to the integral of $f(x) = x\sin x$ over $[1, 9]$.

## 2 Drawing the vector field of an ordinary differential equation of order one

### 2.1 Description

Let us consider the differential equation

$$\frac{dy}{dx} = f(x, y). \tag{2}$$

At each point $(x_0, y_0)$ in the domain $D$ of $f$, we will put a vector $\mathbf{v}$ with slope $k = f(x_0, y_0)$. If $y(x_0) = y_0$, then $k$ is the slope of the tangent to the solution curve $y = y(x)$ of (2) at $(x_0, y_0)$. The $\mathbf{v}$'s make a *vector field* and the picture of this field would give us information about the shape of solution curves of (2), even we have not found yet any solution of (2).

The vector field of (2) will be depicted on a finite grid of points in $D$. This grid is made of lines, paralell to the axes $Ox$ and $Oy$. The intersectional points of those lines are called *grid points* and often indexed by $(x_i, y_j)$, $i = 0, \ldots, N_x$, $j = 0, \ldots, N_y$. For convenience, we will use polar coordinate to locate the terminal point $(x, y)$ of a field vector, with the initial point at the grid point $(x_i, y_j)$. Then, we can write

$$x = x_i + r\cos\varphi,$$
$$y = y_j + r\sin\varphi.$$

Because $k = f(x_i, y_j) = \tan\varphi$ is finite, we may take $-\pi/2 < \varphi < \pi/2$. From $\sin^2\varphi + \cos^2\varphi = 1$ and $\sin\varphi = k\cos\varphi$, we derive

$$\cos\varphi = \frac{1}{\sqrt{1+k^2}}, \quad \sin\varphi = \frac{k}{\sqrt{1+k^2}}.$$

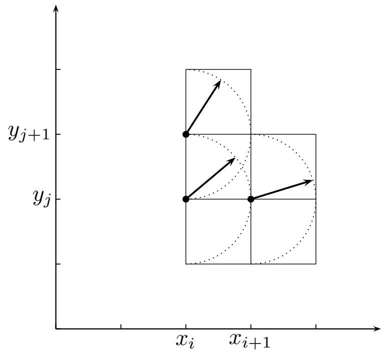


Figure 5: Field vectors on a grid.

The field vectors should all have the same magnitude and we choose here that length to be 1/2, that means $r = 1/2$. Thus, vectors on the grid have their initial points and terminal ones as

$$(x_i, y_j), \quad \Big(x_i + \frac{1}{2}\cos\varphi, y_j + \frac{1}{2}\sin\varphi\Big).$$

Of macros in `PSTricks` to draw lines, we select `\parametricplot`[1] for its fitness. We immetiately have the simple parameterization of the vector at the grid point $(x_i, y_j)$ as

$$x = x_i + \frac{t}{2}\cos\varphi = x_i + \frac{t}{2\sqrt{1+k^2}},$$
$$y = y_j + \frac{t}{2}\sin\varphi = y_j + \frac{tk}{2\sqrt{1+k^2}},$$

where $t$ goes from $t = 0$ to $t = 1$, along the direction of the vector. The macro `\parametricplot` has the syntax as

`\parametricplot[`*settings*`]{`$t_{\min}$`}{`$t_{\max}$`}{`$x(t)$`|`$y(t)$`}`,

where we should use the option `algebraic` to make the declaration of $x(t)$ and $y(t)$ simpler with `ASCII` code.

From the above description of one field vector, we go to the one of the whole vector field on a grid belonging to the domain $R = \{(x, y)\colon a \le x \le b,\ c \le y \le d\}$. To determine the grid, we confine grid points to the range

$$a \le x_i \le b, \quad c \le y_j \le d. \tag{3}$$

With respect to the indices $i$ and $j$, we choose initial values $x_0 = a$ and $y_0 = c$, with increments $\Delta x = \Delta y = \delta$, corresponding to the length of vectors and the distance between grid points as indicated in Figure 5. Thus, to draw vectors at grid points $(x_i, y_j)$, we need two loops for $i$ and $j$, with $0 \le i \le \lfloor m/\delta \rfloor$, $0 \le j \le \lfloor n/\delta \rfloor$, where $m = b - a$, $n = d - c$. Apparently, these two loops are nested `\multido`s, with variable declaration for each loop as follows

$$\texttt{\textbackslash nx} = \text{initial value} + \text{increment} = x_0 + \Delta x,$$
$$\texttt{\textbackslash ny} = \text{initial value} + \text{increment} = y_0 + \Delta y.$$

Finally, we will replace `\nx`, `\ny` by $x_i$, $y_j$ in the below calling sequence for simplicity.

[1]This macro is of ones, often added and updated in the package `pstricks-add`, the authors: Dominique Rodriguez (`dominique.rodriguez@waika9.com`), Herbert Voß (`voss@pstricks.de`).

Thus, the main procedure to draw the vector field of the equation (2) on the grid (3) is

$$\texttt{\textbackslash multido}\{y_j = y_0 + \Delta y\}\{\lfloor n/\delta \rfloor\}\Big\{\texttt{\textbackslash multido}\{x_i = x_0 + \Delta x\}\{\lfloor m/\delta \rfloor\}$$
$$\Big\{\texttt{\textbackslash parametricplot}\texttt{[}\mathit{settings}\texttt{]}\{0\}\{1\}\Big\{x_i + \frac{t}{2\sqrt{1+\left[f(x_i,y_j)\right]^2}} \Big|\ y_j + \frac{tf(x_i,y_j)}{2\sqrt{1+\left[f(x_i,y_j)\right]^2}}\Big\}\Big\}$$

where we at least use `arrows=->` and `algebraic` for *settings*.

We can combine the steps mentioned above to define a drawing procedure, say `\vecfld`, that consists of main parameters in the order as `\nx=`$x_0 + \Delta x$, `\ny=`$y_0 + \Delta y$, $\lfloor m/\delta \rfloor$, $\lfloor n/\delta \rfloor$, $\delta$ and $f($`\nx`$,$ `\ny`$)$. We may change these values to modify the vector field or to avoid the vector intersection. Such a definition is suggested to be

```
\def\vecfld#1#2#3#4#5#6{%
\multido{#2}{#4}{\multido{#1}{#3}
{\parametricplot[algebraic,arrows=->,linecolor=red]{0}{1}
{\nx+((#5)*t)*(1/sqrt(1+(#6)^2))|\ny+((#5)*t)*(1/sqrt(1+(#6)^2))*(#6)}}}}
```

## 2.2 Examples

Firstly, we consider the equation that describes an object falling in a resistive medium:

$$\frac{dv}{dt} = 9.8 - \frac{v}{5}, \tag{4}$$

where $v = v(t)$ is the speed of the object in time $t$. In Figure 6, the vector field of (4) is given on the grid $R = \{(t, y)\colon 0 \le t \le 9,\ 46 \le v \le 52\}$, together with the graph of the equilibrium solution $v = 49$.

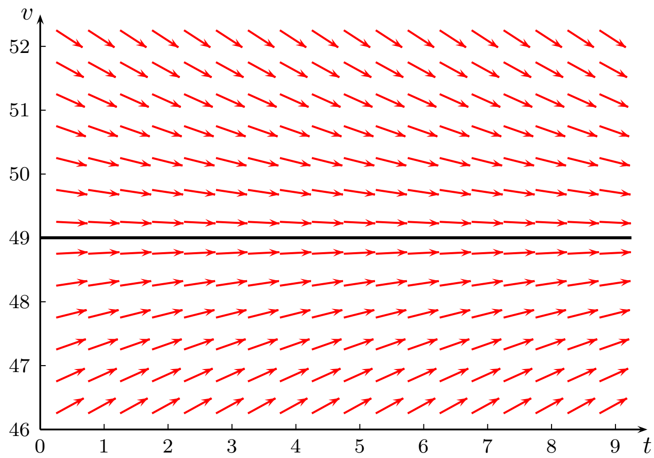


Figure 6: The vector field of (4).

Figure 6 is made of the following LaTeX code:

```
\begin{pspicture}(0,46)(9.5,52.5)
\vecfld{\nx=0.25+0.50}{\ny=46.25+0.50}{18}{12}{0.5}{9.8-0.2*\ny}
\psplot[algebraic,linewidth=1.2pt]{0}{9}{49}
\psaxes[Dy=1,Dx=1,Oy=46]{->}(0,46)(0,46)(9.5,52.5)
\rput(9.5,45.8){$t$}\rput(-0.2,52.5){$y$}
\end{pspicture}
```

Let us next consider the problem

$$\frac{dy}{dx} = x + y, \quad y(0) = 0. \tag{5}$$

It is easy to check that $y = e^x - x - 1$ is the unique solution to the problem (5). We now draw the vector field of (5) and the solution curve[2] on the grid $R = \{(x, y)\colon 0 \le x \le 3,\, 0 \le y \le 5\}$ in Figure 7.

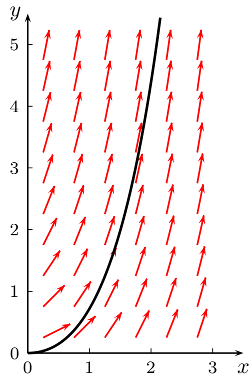


Figure 7: The vector field of (5).

We then go to the logistic equation, which is chosen to be a model for the dependence of the population size $P$ on time $t$ in Biology:

$$\frac{dP}{dt} = kP\Big(1 - \frac{P}{M}\Big), \tag{6}$$

where $k$ and $M$ are constants, respectively various to selected species and environment. For specification, we take, for instant, $k = 0.5$ and $M = 100$. The right hand side of (6) then becomes $f(t, P) = 0.5\,P(1 - 0.01\,P)$. In Figure 8, we draw the vector field of (6) on the grid $R = \{(t, P)\colon 0 \le t \le 10,\, 95 \le P \le 100\}$ and the equilibrium solution curve $P = 100$. Furthermore, with the initial condition $P(0) = 95$, the equation (6) has the unique solution $P = 1900(e^{-0.5t}+19)^{-1}$. This solution curve is also given in Figure 8.

The previous differential equations are all of seperated variable or linear cases that can be solved for closed-form solutions by some simple integration formulas. We will consider one more equation of the non-linear case whose solution can only be approximated by numerical methods. The vector field of such an equation is so useful and we will use the Runge-Kutta curves (of order 4) to add more information about the behaviour of solution curve. Here, those Runge-Kutta curves are depicted by the procedure `\psplotDiffEqn`, also updated from the package `pstricks-add`.

[2] We have used $\mathrm{ch}(1) + \mathrm{sh}(1)$ for the declaration of $e$, natural base of logarithmic function.

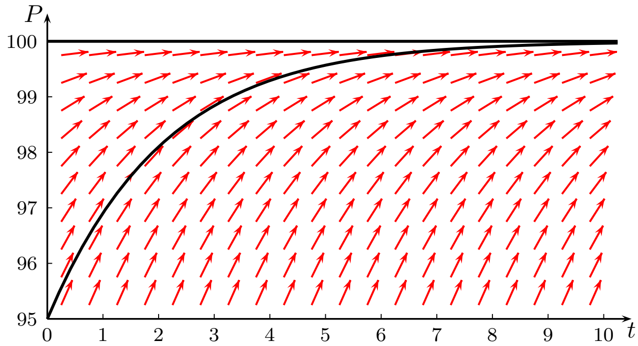


Figure 8: The vector field of (6) with $k = 0.5$ and $M = 100$.

The vector field of the non-linear differential equation

$$\frac{dy}{dx} = y^2 - xy + 1 \tag{7}$$

will be depicted on the grid $R = \{(x, y)\colon -3 \le x \le 3, -3 \le y \le 3\}$ and the solutions of Cauchy problems for (7), corresponding to initial conditions

(i) $y(-3) = -1$,

(ii) $y(-2) = -3$,

(iii) $y(-3) = -0.4$,

will be approximated by the method of Runge-Kutta, with the grid size $h = 0.2$. It is very easy to recognize approximate curves, respective to (i), (ii) and (iii) in Figure 9 below.

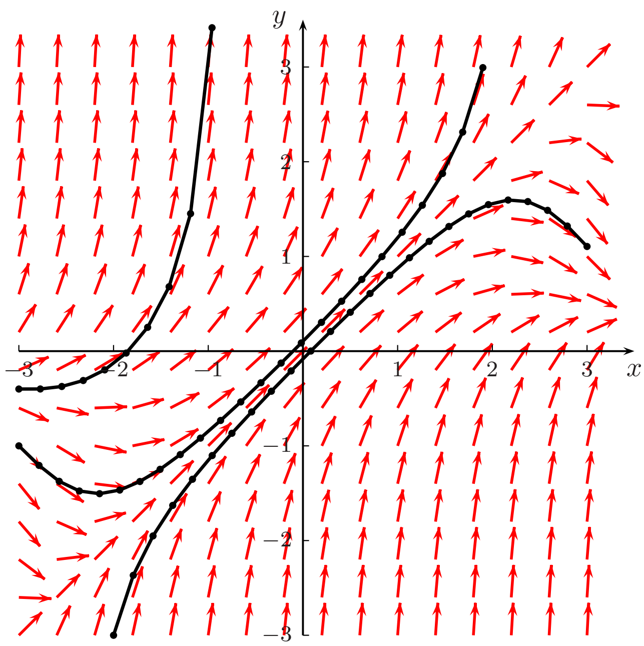


Figure 9: The vector field of (7) and the Runge-Kutta curves.

# 3 Remarks on how to color arrows properly for a vector field

## 3.1 Description

In the `\vecfld` procedure, the command

$$\texttt{\textbackslash parametricplot[}\textit{settings}\texttt{]}\{t_{\min}\}\{t_{\max}\}\{x(t)\,|\,y(t)\} \tag{8}$$

does the two works: drawing the whole oriented line segment and putting the endpoint right after the vector. This blots out the pointy head of arrows and makes field vectors less sharp when being seen closely. However, there is no problem with the procedure if we just want a monochrome vector field. But, in case of using arrows with their various color shades, we should use an independent procedure with options to draw a color arrow. For such a procedure, the command `\psline` could be the best choice. We just call it with two argument points, which are extracted from the curve produced by the command `\parametricplot`.

To modify the `\vecfld` procedure, from the above consideration, we might take the command `\curvepnodes` in the package `pst-node`[3] to extract points from a curve $(x(t), y(t))$ given in the algebraic form. Because we only need the two ending points of the curve, we can use

$$\texttt{\textbackslash curvepnodes[algebraic,plotpoints=2]\{0\}\{1\}}\{x(t)\,|\,y(t)\}\texttt{\{P\}}, \tag{9}$$

where `P` is a name of the root of nodes and we just get the two nodes `P0`, `P1` when executing this command. Then, the corresponding vector is drawn by the command

$$\texttt{\textbackslash psline[linecolor=}\textit{settings}\texttt{]\{->\}(P0)(P1)} \tag{10}$$

The command (8) may be replaced by the two ones (9) and (10), and we obtain the arrows whose heads are now sharper.

The remaining problem is how to appropriately make *settings* in (10) to bring out a vector field. Obviously, *settings* should be various color shades according to slope of vectors. In Subsection 2.1, we know for the equation (2) that $f(x_i, y_j)$ is right the slope of field vectors at grid points $(x_i, y_j)$, and we will divide these slopes into some number of scales, corresponding to the degree of color shades. Here, we confine our interest to a continuous function $f(x, y)$ in two independent variables on the domain $R = \{(x, y)\colon a \le x \le b,\ c \le y \le d\}$ and choose the scale of 10 degrees. This number of degrees can be changed to any positive integer.

According to the input data from the differential equation (2), the set $R$ and the grid points on it and the value $M = \max\{|f(x_i, y_j)|\colon 0 \le i \le \lfloor m/\Delta x\rfloor,\ 0 \le j \le \lfloor n/\Delta y\rfloor\}$, where $m = b - a$ and $n = d - c$, we can now define the degree of color shade for each arrow in our vector field. It should be an integer $n_{ij}$ such that $n_{ij} = \lfloor 10|f(x_i, y_j)|/M\rfloor$, that is

$$n_{ij}M \le 10|f(x_i, y_j)| < (n_{ij} + 1)M. \tag{11}$$

For finding such an integer, in TEX codes, we need one `\newcount` for it and two `\newdimen` for $f(x_i, y_j)$ and intermediate values to be compared with $|f(x_i, y_j)|$. For more explanation, let us begin with settings `\newcount\intg` (referring (ref.) to "integer"), `\newdimen\slope` (ref. to "slope") and `\newdimen\interm` (ref. to "intermediate values"). Then, the integer $n_{ij}$ at stage $(i, j)$ within the two loops `\multido` can be defined by the recursive macro `\fintg` (ref. to "find the integer") as follows

[3] Package authors: Timothy Van Zandt (`tvz@econ.insead.fr`), Michael Sharpe (`msharpe@euclid.ucsd.edu`) and Herbert Voß (`hvoss@tug.org`).

```
\def\fintg{\interm=Mpt \interm=\intg\interm%
    \ifdim\ifdim\slope<0pt -\fi\slope>\interm \advance\intg by 1\fintg\fi}
```

where `M` and `\slope` are holding the values $M$ and $f(x_i, y_j)$, respectively. Note that, before running our macro, `\slope` should be multiplied by 10 with the assignment `\slope=10\slope`, as defined in (11). Besides, by simulating the expression of $f(x, y)$, the calculation of $f(x_i, y_j)$ should be declared with operations on `\newcounts` and `\newdimens`. Then, the integer $n_{ij}$, which is found at stage $(i, j)$, should take its degree, say $k$, from 0 to 10 by its value, suitably associated to the command `\psline[linecolor=red!case-k]{->}(P0)(P1)`. Here, we choose `red` for the main color (it can be changed, of course), and `case-k` will be replaced with an appropriate percentage of `red`. Finally, making such a color scale is local and relative, so we can use one more parameter in the procedure to adjust color shades. The old procedure takes 6 parameters and the new one will take two more parameters: one for a way of computing $f(x_i, y_j)$ and the other for adjusting color shades.

Let us take some examples on how to compute $f(x_i, y_j)$ by TEX codes or by the commands from the package `calculator`[4]. For a simple polynomial $f(x, y)$, computing $f(x_i, y_j)$ by TEX codes might be facile. Because `\nx` and `\ny` are respectively holding the values of $x_i$ and $y_j$, we need the two corresponding dimensions `\newdimen\fx` and `\newdimen\fy` to take these values. By assigning `\fx=\nx pt\fy=\ny pt`, we compute $f($`\nx`$,$ `\ny`$)$ and assign its value to `\slope`. The declaration of calculations for some cases of $f(x, y)$ is given in the following table.

| $f(x,y)$ | TEX codes for computing $f($`\nx`, `\ny`$)$ |
|---|---|
| $x + y$ | `\advance\slope by \fx \advance\slope by \fy` |
| $1 - xy$ | `\advance\slope by -\decimal\fx\fy \advance\slope by 1pt` |
| $y(3 - y)$ | `\advance\slope by -\decimal\fy\fy \advance\slope by 3\fy` |
| $y^2 - xy$ | `\advance\slope by \decimal\fy\fy \advance\slope by -\decimal\fx\fy` |

In the table, the command `\decimal`, which is quotative from [5] for producing decimal numbers from dimensions, is put in the preamble using a definition as

```
\def\xch{\catcode`\p=12 \catcode`\t=12}\def\ych{\catcode`\p=11 \catcode`\t=11}
      \xch \def\dec#1pt{#1}\ych \def\decimal#1{\expandafter\dec \the#1}
```

For a transcendental or rational function $f(x, y)$, we should use the package `calculator` for computing $f(x_i, y_j)$. The following table shows how to perform the calculations.

| $f(x,y)$ | The commands from the package `calculator` for computing $f($`\nx`, `\ny`$)$ |
|---|---|
| $\sin(y - x)$ | `\SUBTRACT{\ny}{\nx}{\sola}\SIN{\sola}{\solb}\slope=\solb pt` |
| $2xy/(1 + y^2)$ | `\SUMfunction{\ONEfunction}{\SQUAREfunction}{\Fncty}`<br>`\Fncty{\ny}{\soly}{\Dsoly}\DIVIDE{\Dsoly}{\soly}{\tempa}`<br>`\MULTIPLY{\nx}{\tempa}{\tempb}\slope=\tempb pt` |

From the old macro `\vecfld`, we will construct the new one `\vecfldnew` by adding up to the former the two parameters as described above. According to the description of new parameters and of known ones, the calling sequence of `\vecfldnew` may have the form of

`\vecfldnew`{`\nx=` $x_0 + \Delta x$}{`\ny=` $y_0 + \Delta y$}{$n_x$}{$n_y$}{$\ell$}{$f($`\nx`, `\ny`$)$}{TEX codes}{$n_a$}

[4]Package author: Robert Fuster (`rfuster@mat.upv.es`).

where $n_a$ is an estimate value for $M$ and can be adjusted to be greater or less than $M$. This flexible mechanism might be to increase or decrease the degree of color shades. Finally, `\intg` and `\slope` should be reset to zero at the end of each stage. Now, all materials to make the new macro are ready, and a definition for it is suggested to be

```
\def\vecfldnew#1#2#3#4#5#6#7#8{%
\newcount\intg \newdimen\slope \newdimen\interm \newdimen\fx \newdimen\fy
\def\fintg{\interm=#8 \interm=\intg\interm%
    \ifdim\ifdim\slope<0pt -\fi\slope>\interm \advance\intg by 1\fintg\fi}
\multido{#2}{#4}
{\multido{#1}{#3}
{\curvepnodes[algebraic,plotpoints=2]{0}{1}
{\nx+((#5)*t)*(1/sqrt(1+(#6)^2))|\ny+((#5)*t)*(1/sqrt(1+(#6)^2))*(#6)}{P}
#7\slope=10\slope \fintg \ifnum\intg>10\psline[linecolor=red]{->}(P0)(P1)
\else\ifnum\intg=0\psline[linecolor=red!5]{->}(P0)(P1)
\else\multiply\intg by 10\psline[linecolor=red!\the\intg]{->}(P0)(P1)\fi\fi
\intg=0\slope=0pt
}}}
```

If we predefine some scale of degrees, instead of the code `\ifnum\intg>10...\fi\fi`, the structure `\ifcase` can be used as

```
\ifcase\intg
\psline[linecolor=red!5]{->}(P0)(P1)\or
\psline[linecolor=red!10]{->}(P0)(P1)\or
⋮
\psline[linecolor=red]{->}(P0)(P1)\fi
```

### 3.2 Examples

The first example is given with the two $n_a$s to see how different the color shades are between the two cases. The left vector field in Figure 10 is made of the calling sequence

```
\vecfldnew{\nx=-2.00+0.3}{\ny=-2.00+0.3}{14}{14}{0.3}{(\nx)-2*(\ny)}
{\fy=\ny pt \fx=\nx pt \advance\slope by -2\fy \advance\slope by \fx}{9pt}
```

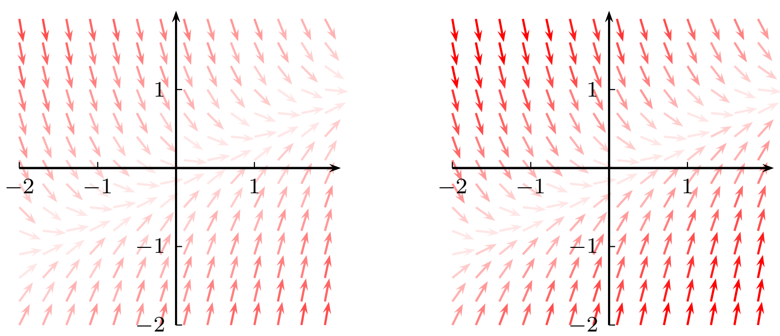


Figure 10: The vector fields of the equation $y' = x - 2y$ with $n_a =$ `9pt` (the left) and $n_a =$ `5pt` (the right)

In Figure 11, the vector fields of the equations $y' = y - x$ and $y' = x(2 - y)$ are respectively drawn by the calling sequences

```
\vecfldnew{\nx=-3.00+0.4}{\ny=-3.00+0.4}{15}{15}{0.35}{(\ny)-(\nx)}
{\fy=\ny pt \fx=\nx pt \advance\slope by -\fx \advance\slope by \fy}{5pt}
```

and

```
\vecfldnew{\nx=-3.00+0.4}{\ny=-3.00+0.4}{15}{15}{0.35}{(\nx)*(2-(\ny))}
{\fy=\ny pt \fx=\nx pt \advance\slope by -\decimal\fx\fy
\advance\slope by 2\fx}{6pt}
```

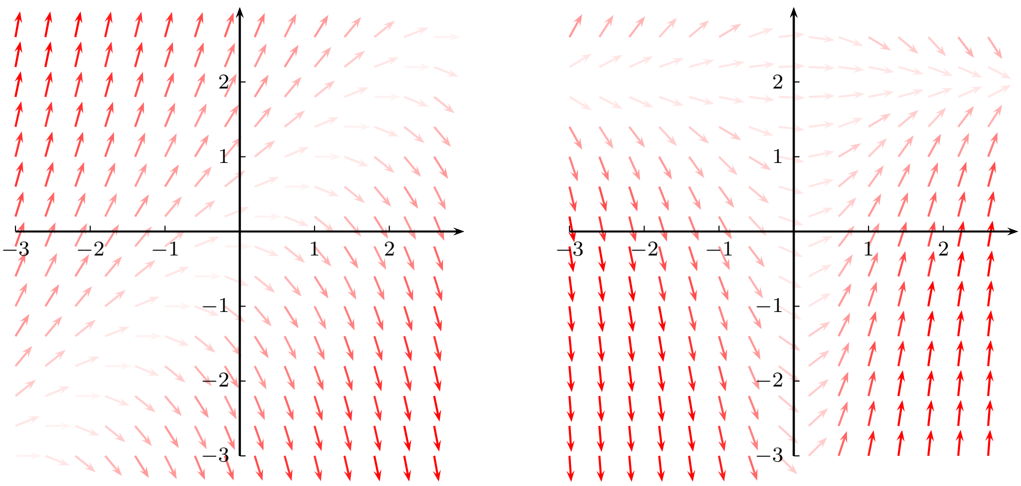


Figure 11: The vector fields of the equation $y' = y - x$ (the left) and $y' = x(2 - y)$ (the right).

Finally, we consider two more examples on vector fields of differential equations $y' = f(x, y)$ containing trigonometric or rational functions on the right side. The calling sequences

```
\vecfldnew{\nx=-3.00+0.4}{\ny=-3.00+0.4}{15}{15}{0.35}{sin(\nx)*cos(\ny)}
{\SIN{\nx}{\tmpa}\COS{\ny}{\tmpb}\MULTIPLY{\tmpa}{\tmpb}{\tmpc}
\slope=\tmpc pt}{0.6pt}
```

and

```
\vecfldnew{\nx=-3.00+0.3}{\ny=-3.00+0.3}{20}{20}{0.3}{2*(\nx)*(\ny)/(1+(\ny)^2)}
{\SUMfunction{\ONEfunction}{\SQUAREfunction}{\Fncty}\Fncty{\ny}{\soly}{\Dsoly}
\DIVIDE{\Dsoly}{\soly}{\tempa}\MULTIPLY{\nx}{\tempa}{\tempb}
\slope=\tempb pt}{2.5pt}
```

respectively result in the vector field on the left and on the right in Figure 12.

## Acknowledgment

I am very grateful to

- Timothy Van Zandt, Herbert Voß, Dominique Rodriguez and Michael Sharpe for helping me with their great works on `PSTricks`.
- Hàn Thế Thành for helping me with his pdfLaTeX program.
- Robert Fuster for his very useful package `calculator`.

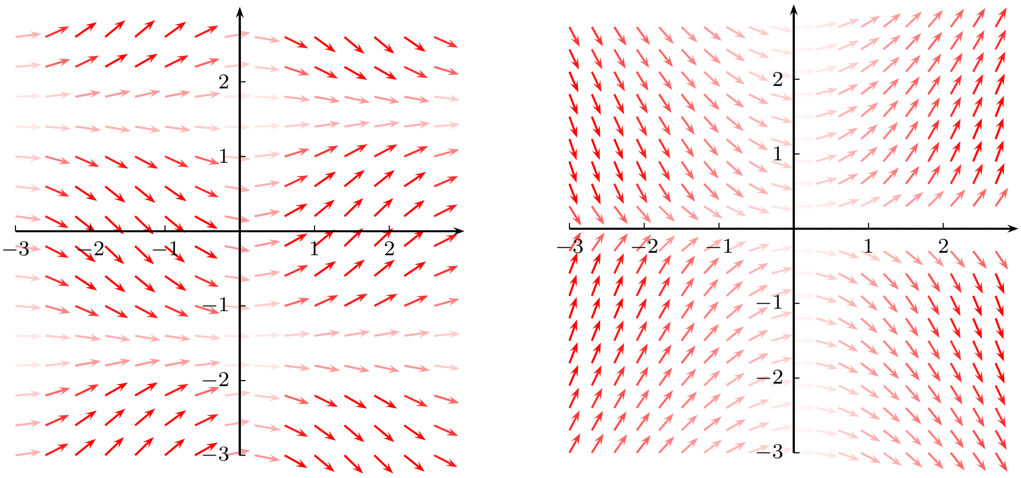


Figure 12: The vector fields of the equation $y' = \sin(x)\cos(y)$ (the left) and $y' = 2xy/(1+y^2)$ (the right).

## References


[1] Dominique Rodriguez, Michael Sharpe & Herbert Voß. *pstricks-add: Additional Macros for PSTricks*. Version 3.60, http://ctan.org/tex-archive/graphics/pstricks/contrib, 2013

[2] Timothy Van Zandt, Michael Sharpe & Herbert Voß. *pst-node: Nodes and node connections*. Version 1.29, http://ctan.org/tex-archive/graphics/pstricks/contrib, 2013

[3] Helmut Kopka & Patrick W. Daly. *Guide to LaTeX*. Addison-Wesley, Fourth Edition, 2004, ISBN 0321173856

[4] Timothy Van Zandt. *User's Guide*. Version 1.5, http://ctan.org/tex-archive/graphics/pstricks/base, 2007

[5] Eitan M. Gurari. *Writing With TeX*, McGraw-Hill, Inc., 1994, ISBN 0-07-025207-6

[6] Robert Fuster. *calculator-calculus: Scientific Calculations With LaTeX*. Version 1.0a, http://ctan.org/tex-archive/macros/latex/contrib/calculator, 2012