

---

## Ebooks and paper sizes: Output routines made easier

Boris Veytsman and Michael Ware

### Abstract

The idea of a book being a collection of pages is so ingrained that modern electronic book readers often try to faithfully reproduce this feature—up to elaborate simulations of page turns. However, traditional pages are not necessary and often inconvenient in electronic books. It is often easier to scroll the text than to turn the pages, and text reflowing makes the use of folios a rather strange way to refer to the position inside a text.

We argue that it is more natural to paginate electronic books according to their logical structure, when a “page” corresponds to a sectional unit of the book. This leads to rather long pages, with the height of the page depending on the length of the corresponding unit.

We discuss how to implement these pages in  $\text{\TeX}$  and provide a basic introduction to output routines in  $\text{\TeX}$  for a beginning  $\text{\TeX}$ nician. We also provide exercises for a slightly more advanced reader.

### 1 Introduction

$\text{\TeX}$ -based systems have been creating high-quality electronic books (ebooks) for decades, with PDF becoming the dominant format in recent years. While there are many macro packages that optimize  $\text{\TeX}$ 's PDF output for electronic reading (with links, etc.), the basic paradigm of  $\text{\TeX}$ -produced ebooks is still very much tied to the ideas of a physical book—the document is formatted into a series of identically-sized pages with the position of floating environments and graphics adjusted so as to avoid the page breaks. This type of ebook works well both in print and on screens that are similar in size to a piece of paper. However, the proliferation of mobile devices with small screens presents a new opportunity and challenge for ebook creators.

Typical PDF pages cannot be displayed in their entirety on small screens. To read a full-page document on a smartphone, one has to zoom in until the text is sufficiently large, and then pan around through the document. This makes for a difficult reading experience, and has caused ebook creators to largely abandon fixed-layout schemes like PDF in favor of reflowable formats such as HTML. This “just-in-time” layout strategy allows the page layout software to adjust the font size to a readable level and the line width so that one only has to scroll in one direction to view all of the text.

Current reflow schemes work well for documents that are primarily text-based, but they typically have limited support for technical documents with an abundance of equations and figures. Since equations and figures are a critical part of a technical text, the current generation of ebook readers does not provide a workable solution for most technical documents. Some progress has been made in browser-based HTML with technologies such as MathJax (see <http://www.mathjax.org/>) and we may eventually have a dedicated ebook reader that uses  $\text{\TeX}$  as its layout engine (Bazargan, 2009). However, it seems likely that we will continue to use PDF documents for some time for technical ebooks, so the question remains: how can  $\text{\TeX}$ -created documents be optimized for a smooth ebook reading experience?

To make these issues more concrete, consider the layout of the textbook pages shown in Fig. 1. This is a typical textbook layout, with figures in the margins near the referring text, footnotes at the bottom of each page, and headings atop each page to give the reader information about their location in the text. These features make for an easy reading experience on paper or on a large screen, but can get in the way when viewed on even a moderately sized e-reader such as an iPad. In this reading environment, a reader needs to scroll horizontally to view figures, and the flow of the text is interrupted on each page to show the footnotes and header information.

Since a PDF document has a fixed layout,<sup>1</sup> it is necessary to choose the page size parameters in advance in a way that works for the intended screen size. An obvious first approach is to make the  $\text{\TeX}$  page size match the size of the intended reading device and minimize the margins around the text (Cheswick, 2011). However, this has some drawbacks. As the page size gets smaller, it becomes increasingly difficult to lay out non-text elements such as figures and larger equations. Since the pages can be quite small, there are a lot of page breaks,  $\text{\TeX}$ 's float placement routines tend to leave many pages with awkward white space, and the floats are far from the referring text.

A better approach is to make the pages fit the target screen width, but be very tall relative to the intended screen size. Each logical division of the text (say, a section) is placed on its own page. In this scenario, figure placement and equation layout are easy for  $\text{\TeX}$ , since the page breaks essentially disappear. The reader can view the full width and scroll vertically through the content of a section, and flipping

---

<sup>1</sup> One can reflow the text portion of a document in newer PDF specifications, but images do not reflow.

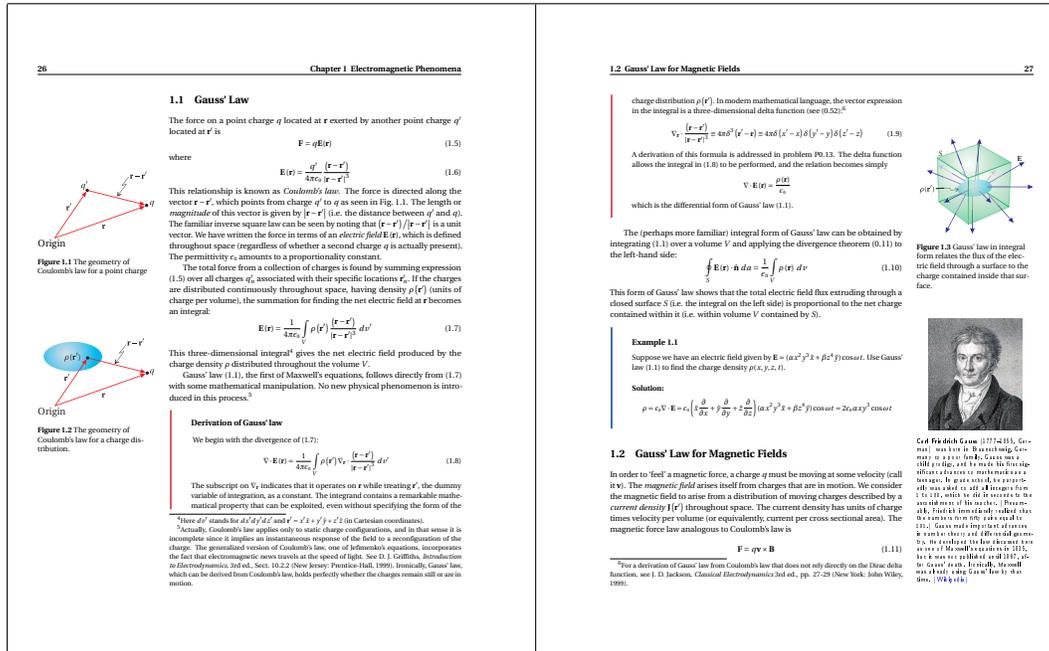


Figure 1: Some typical pages of a textbook formatted for paper printing. (See <http://optics.byu.edu/>)

to the next “page” moves to the next logical division of the text. This layout has the added benefit of getting rid of many old typography problems, such as widows, orphans, badly placed floating material, etc.

However, since the text of each section is of a different length while  $\text{T}_{\text{E}}\text{X}$  maintains a fixed page length, each page has white space at the bottom that a user must scroll through. It would be nice if  $\text{T}_{\text{E}}\text{X}$  provided a way to make each page just the right size for its content, and fortunately it does.

Figure 2 shows the same text as Fig. 1 formatted using the “tall page” approach described above. The figures have been moved inline with the text using some straightforward modifications to the margin figure macros, and the width of the text is now appropriate for full-width viewing on a tablet device. The informational header is still retained at the top of the page and the footnotes appear at the bottom, but they no longer interrupt the flow of the text. Now the reader can view the full content of this section of the textbook simply by scrolling vertically. To view the next section of the book, the reader simply moves to the next “page” of the ebook.

With some planning, it is very reasonable to design a document class that can be converted from a traditional paper layout like Fig. 1 to an ebook layout like Fig. 2 with a single command switch.<sup>2</sup> This allows an author to easily provide multiple layouts

<sup>2</sup> For example, the code that made the example pages in Figs. 1 and 2 is available at [optics.byu.edu](http://optics.byu.edu).

that are appropriate for both paper and on-screen reading. To do this, it is necessary to understand how paper size is treated in  $\text{T}_{\text{E}}\text{X}$ .

## 2 Paper length in $\text{T}_{\text{E}}\text{X}$

It might be surprising for a beginning  $\text{T}_{\text{E}}\text{X}$ nician to learn the extent to which the “classical”  $\text{T}_{\text{E}}\text{X}$  cares about actual paper dimensions: namely, it does not care about them at all — and does not even know about them. Of course, paper dimensions are mentioned in many  $\text{T}_{\text{E}}\text{X}$  macro packages — for example,  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  geometry package (Umeki, 2010), but in many cases they are just used to calculate the dimensions of the text area.

One can argue that this feature corresponds to the rôle of  $\text{T}_{\text{E}}\text{X}$  as a compositor: in a classic printing shop a compositor puts words into a matrix, but the choice of the paper on which the imprint on was done by another artisan. A more prosaic explanation is that the printers used during the time  $\text{T}_{\text{E}}\text{X}$  was written had no means to change the paper dimensions, so it made little sense to control them in a typesetting program.

DVI drivers, however, could deal with paper size, for example, through the use of PostScript commands as arguments of `\specials` in `dvips`. Since `pdftex` is both a  $\text{T}_{\text{E}}\text{X}$  engine and a (PDF) driver, it has commands `\pdfpageheight` and `\pdfpagewidth`, which deal with paper dimensions directly. In this section

**Chapter 1 Electromagnetic Phenomena** 9

**1.1 Gauss' Law**

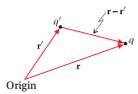
The force on a point charge  $q$  located at  $\mathbf{r}$  exerted by another point charge  $q'$  located at  $\mathbf{r}'$  is

$$\mathbf{F} = q\mathbf{E}(\mathbf{r}) \quad (1.5)$$

where

$$\mathbf{E}(\mathbf{r}) = \frac{q'}{4\pi\epsilon_0} \frac{[\mathbf{r} - \mathbf{r}']}{|\mathbf{r} - \mathbf{r}'|^3} \quad (1.6)$$

This relationship is known as *Coulomb's law*. The force is directed along the vector  $\mathbf{r} - \mathbf{r}'$ , which points from charge  $q'$  to  $q$  as seen in Fig. 1.1. The length or *magnitude* of this vector is given by  $|\mathbf{r} - \mathbf{r}'|$  (i.e. the distance between  $q'$  and  $q$ ). The familiar inverse square law can be seen by noting that  $[\mathbf{r} - \mathbf{r}']/|\mathbf{r} - \mathbf{r}'|^3$  is a unit vector. We have written the force in terms of an *electric field*  $\mathbf{E}(\mathbf{r})$ , which is defined throughout space (regardless of whether a second charge  $q$  is actually present). The permittivity  $\epsilon_0$  amounts to a proportionality constant.

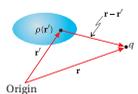


**Figure 1.1** The geometry of Coulomb's law for a point charge

The total force from a collection of charges is found by summing expression (1.5) over all charges  $q'_i$  associated with their specific locations  $\mathbf{r}'_i$ . If the charges are distributed continuously throughout space, having density  $\rho(\mathbf{r}')$  (units of charge per volume), the summation for finding the net electric field at  $\mathbf{r}$  becomes an integral:

$$\mathbf{E}(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \int_V \rho(\mathbf{r}') \frac{[\mathbf{r} - \mathbf{r}']}{|\mathbf{r} - \mathbf{r}'|^3} dV' \quad (1.7)$$

This three-dimensional integral<sup>6</sup> gives the net electric field produced by the charge density  $\rho$  distributed throughout the volume  $V$ .



**Figure 1.2** The geometry of Coulomb's law for a charge distribution.

Gauss' law (1.1), the first of Maxwell's equations, follows directly from (1.7) with some mathematical manipulation. No new physical phenomenon is introduced in this process.<sup>5</sup>

**Derivation of Gauss' law**

We begin with the divergence of (1.7):

$$\nabla \cdot \mathbf{E}(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \int_V \rho(\mathbf{r}') \nabla \cdot \frac{[\mathbf{r} - \mathbf{r}']}{|\mathbf{r} - \mathbf{r}'|^3} dV' \quad (1.8)$$

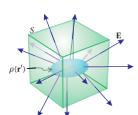
The subscript on  $\nabla$  indicates that it operates on  $\mathbf{r}$  while treating  $\mathbf{r}'$ , the dummy variable of integration, as a constant. The integrand contains a remarkable mathematical property that can be exploited, even without specifying the form of the charge distribution  $\rho(\mathbf{r}')$ . In modern mathematical language, the vector expression in the integrand is a three-dimensional delta function (see (0.52))<sup>8</sup>

$$\nabla \cdot \frac{[\mathbf{r} - \mathbf{r}']}{|\mathbf{r} - \mathbf{r}'|^3} = 4\pi\delta^3(\mathbf{r} - \mathbf{r}') = 4\pi\delta(x' - x)\delta(y' - y)\delta(z' - z) \quad (1.9)$$

A derivation of this formula is addressed in problem P0.13. The delta function allows the integral in (1.8) to be performed, and the relation becomes simply

$$\nabla \cdot \mathbf{E}(\mathbf{r}) = \frac{\rho(\mathbf{r})}{\epsilon_0}$$

which is the differential form of Gauss' law (1.1).



**Figure 1.3** Gauss' law in integral form relates the flux of the electric field through a surface to the charge contained inside that surface.

The (perhaps more familiar) integral form of Gauss' law can be obtained by integrating (1.1) over a volume  $V$  and applying the divergence theorem (0.11) to the left-hand side:

$$\oint_S \mathbf{E}(\mathbf{r}) \cdot \mathbf{n} dA = \frac{1}{\epsilon_0} \int_V \rho(\mathbf{r}) dV \quad (1.10)$$

This form of Gauss' law shows that the total electric field flux extruding through a closed surface  $S$  (i.e. the integral on the left side) is proportional to the net charge contained within it (i.e. within volume  $V$  contained by  $S$ ).

**Example 1.1**

Suppose we have an electric field given by  $\mathbf{E} = (ax^2y^3z + \beta z^2y) \cos\omega t$ . Use Gauss' law (1.1) to find the charge density  $\rho(x, y, z, t)$ .

**Solution:**

$$\rho = \epsilon_0 \nabla \cdot \mathbf{E} = \epsilon_0 \left( \frac{\partial}{\partial x} (2axy^3z) + \frac{\partial}{\partial y} (3ax^2y^2z) + \frac{\partial}{\partial z} (2\beta z^2y) \right) \cos\omega t = 2\epsilon_0 ax^2y^3 \cos\omega t$$

**Carl Friedrich Gauss** (1777–1855, German) was born in Brunswick, Germany to a poor family. Gauss was a child prodigy, and he made his first significant advances in mathematics as a teenager. In grade school, he reportedly was asked to add all integers from 1 to 100, which he did in seconds to the astonishment of his teacher. (Presumably, Friedrich immediately realized that the numbers from fifty pairs equal to 100.) Gauss made important advances in number theory and differential geometry. He developed the law discussed here as one of Maxwell's equations in 1835, but it was not published until 1867, after Gauss' death. Ironically, Maxwell was already using Gauss' law by that time. (Wikipedia)

<sup>6</sup>Here  $dV'$  stands for  $dV' = dx' dy' dz'$  and  $\mathbf{r}' = x'\mathbf{i} + y'\mathbf{j} + z'\mathbf{k}$  in Cartesian coordinates.

<sup>5</sup>Actually, Coulomb's law applies only to static charge configurations, and in that sense it is incomplete since it implies an instantaneous response of the field to a reconfiguration of the charge. The generalized version of Coulomb's law, one of Jefimenko's equations, incorporates the fact that electromagnetic news travels at the speed of light. See D. J. Griffiths, *Introduction to Electrodynamics*, 3rd ed., Sect. 10.2.2 (New Jersey: Prentice-Hall, 1999). Ironically, Gauss' law, which can be derived from Coulomb's law, holds perfectly whether the charges remain still or are in motion.

<sup>8</sup>For a derivation of Gauss' law from Coulomb's law that does not rely directly on the Dirac delta function, see J. D. Jackson, *Classical Electrodynamics* 3rd ed., pp. 27–29 (New York: John Wiley, 1999).

Figure 2: The content from the pages in Fig. 1 formatted in an “ebook friendly” layout.

```

\size=500cm
\pdfpageheight=500cm
\hrule
\vskip 1in
\centerline{\bf A SHORT STORY}
\vskip 6pt
\centerline{\sl by A. U. Thor}
\vskip .5cm
Once upon a time, in a distant galaxy called
"0"o c, there lived a computer named
R.~J. Drofnats.

Mr.~Drofnats---or ‘R. J.,’ as
he preferred to be called---was happiest when
he was at work typesetting beautiful documents.
\vskip 1in
\hrule
\vfill\ejct
\bye

```

Figure 3: Long page in plain T<sub>E</sub>X: A simple example.

we discuss how to employ these commands to set up page height for electronic books.

We start from a simple example shown on Figure 3. It differs from the classical T<sub>E</sub>X story (Knuth, 1994) by two lines: the command `\size=500cm` tells T<sub>E</sub>X that the “galley” is very long, and the command `\pdfpageheight=500cm` sets the paper height to the same value.

**Exercise 1:** Actually it is a rather poor idea to set both these lengths to the same value. Why?

When we compile this file with `pdftex`, we get a very long page with a lot of white space at the bottom. As discussed in the previous section, we would like to get rid of the extra white space and make the page size fit the text. To do this we need to set `\pdfpageheight` dynamically. For this would like to know the height of the text at the moment the page is “shipped out”. This means changing the output routine of T<sub>E</sub>X.

### 3 Output routines

As T<sub>E</sub>X processes a document, it arranges a block of material (text, equations, figures, etc.) until the block’s size is near a predetermined target (stored in `\pagegoal`), or it runs out of material. At this point, T<sub>E</sub>X hands the collected material off to an output routine that does some final manipulation and adds page numbers, headers, footers, etc. to the page. The page is then shipped out to the final document, and T<sub>E</sub>X moves on to the next page.

Traditionally, output routines are considered one of the hardest parts of T<sub>E</sub>X—probably because

---

```
\output={\shipout\box255}
```

---

**Figure 4:** World’s simplest output routine.

---

```
\output={%
  \pdfpageheight=\pagetotal
  \advance\pdfpageheight by 2in
  \shipout\box255}
\vsizel=500cm
\hrule
\vskip 1in
\centerline{\bf A SHORT STORY}
\vskip 6pt
...
```

---

**Figure 5:** Modified plain T<sub>E</sub>X example.

they *are* hard. This section is intended to be a gentle introduction. A more rigorous introduction can be found in the book (Eijkhout, 2007), and the comprehensive tutorial in the papers (Salomon, 1990a; Salomon, 1990b; Salomon, 1990c).

For our purposes, we need two facts about output routines: first, that the output page is contained in box 255, and second, that the current text height is contained in the value `\pagetotal`. The first fact leads to the world’s simplest output routine (Figure 4). The second one suggests the following modification of this routine: let us set up the paper height to `\pagetotal` plus 2 inches to allow for 1 inch top and bottom margins. This leads to the modified example shown on Figure 5. This example produces a page of the height determined by the material on the page—exactly what we wanted!

Of course our output routines are rather toy-like. A self-respecting output routine should include headers, footers, folios, footnotes, etc. We can take plain T<sub>E</sub>X output routine and patch it with the code setting `\pdfpageheight`, but here we shall leave this as an exercise to the reader.

**Exercise 2:** Try to patch the plain T<sub>E</sub>X output routine in the way described above.

Instead we turn to the L<sup>A</sup>T<sub>E</sub>X output routine. It is very powerful and complex, and may look intimidating for a beginning (or expert) T<sub>E</sub>Xnician. Fortunately, there are packages that allow for patching this routine without looking too deeply into its code. In this paper we will use one such package, `everyshi` (Schröder, 2001). This package allows one to *add* code to the L<sup>A</sup>T<sub>E</sub>X output routine. Thus we can write down our patch as shown on Figure 6.

Unfortunately, this solution has a flaw. To see it, try to compile a file shown on Figure 7. It has

---

```
\textheight500cm
\usepackage{everyshi}
\EveryShipout{%
  \pdfpageheight=\pagetotal
  \advance\pdfpageheight by 2\topmargin
  \advance\pdfpageheight by 2in}
```

---

**Figure 6:** L<sup>A</sup>T<sub>E</sub>X output routine patched.

---

```
\documentclass{article}
\usepackage{everyshi,lipsum}
\pagestyle{empty}
\textheight500cm
\EveryShipout{%
  \pdfpageheight=\pagetotal
  \advance\pdfpageheight by 2\topmargin
  \advance\pdfpageheight by 2in}
\begin{document}

\lipsum[3-5]
\pagebreak

This line has a footnote\footnote{\lipsum[6-8]}.
\lipsum[1]

And this line too\footnote{\lipsum[12]}.
\pagebreak

\end{document}
```

---

**Figure 7:** A L<sup>A</sup>T<sub>E</sub>X file with footnotes.

rather lengthy footnotes—and the output routine patch in Fig. 6 cuts them off!

What happens to the footnotes? It turns out that `\pagetotal` is the height of the text part of the page, and footnotes are not accounted for here. To resolve this issue, we note that the dimension `\pagegoal` keeps the height of the page *minus* the height of footnotes: this is the height of the page “accessible for the text”. Therefore the height of the footnotes is `\textheight minus \pagegoal`. Once we understand this, we can easily modify our code as in Figure 8 to account for the footnotes.

**Exercise 3:** Compile the code in Figure 8 and check that it solves our problem.

**Exercise 4:** Ross Moore suggested a different solution to the problem of footnotes, based on the fact that minipages include the footnotes’ height into the total height. Try to implement it.

**Exercise 5:** Change `\pagebreak` to `\newpage` in Figure 8. What happens? Why?

---

```

\documentclass{article}
\usepackage{everyshi,lipsum}
\pagestyle{empty}
\textheight500cm
\EveryShipout{%
  \pdfpageheight=\pagetotal
  \advance\pdfpageheight by 2in
  \advance\pdfpageheight by 2\topmargin
  \advance\pdfpageheight by \textheight
  \advance\pdfpageheight by -\pagegoal}
\begin{document}

\lipsum[3-5]
\pagebreak

This line has a footnote\footnote{\lipsum[6-8]}.
\lipsum[1]

And this line too\footnote{\lipsum[12]}.
\pagebreak

\end{document}

```

---

**Figure 8:** A corrected L<sup>A</sup>T<sub>E</sub>X output routine.

#### 4 Conclusion

Using the procedure described above, one can create well-formatted content with essentially no superfluous white space. This seems an excellent approach for producing technical electronic content where on-the-fly layout engines will not work.

Finally, we note that the current state of PDF viewers on mobile devices still leaves something to be desired. For example, on the iPad, we have not found an app that properly handles PDF links. Also, the current generation of apps have limited ability to control the zoom state of a page. Some require one to scroll through a PDF vertically, and apps that let you flip through pages with gestures typically scale each page to fit entirely on the screen; there were no options for fixing the page width to fit. For a long page, this produces a thin strip of unreadable text which then must be manually zoomed to read. Nevertheless, these annoyances seem likely to be fixed as new apps are continually produced.

#### Acknowledgements

We are grateful to Frank Mittelbach who found an error in our code, to Ross Moore who suggested an alternative solution for the problem of footnotes and to other participants of TUG 2011 for many interesting suggestions and comments.

#### References

- Bazargan, Kaveh. “T<sub>E</sub>X as an ebook reader”. *TUGboat* **30**(2), 272–73, 2009. <http://river-valley.tv/tex-as-an-ebook-reader>, <http://tug.org/TUGboat/30-2/tb95bazargan.pdf>.
- Cheswick, William. “T<sub>E</sub>X and the iPad”. *TUGboat* **32**(2), 158–163, 2011. <http://river-valley.tv/tex-and-the-ipad>, <http://tug.org/TUGboat/32-2/tb102cheswick.pdf>.
- Eijkhout, Victor. *T<sub>E</sub>X by Topic*. Lulu, 2007. <http://eijkhout.net/texbytopic/texbytopic.html>.
- Knuth, Donald Ervin. *The T<sub>E</sub>Xbook*. Computers & Typesetting A. Addison-Wesley Publishing Company, Reading, MA, 1994. Illustrations by Duane Bibby.
- Salomon, David. “Output Routines: Examples and Techniques. Part I: Introduction and Examples”. *TUGboat* **11**(1), 69–85, 1990a.
- Salomon, David. “Output Routines: Examples and Techniques. Part II: OTR Techniques”. *TUGboat* **11**(2), 212–236, 1990b.
- Salomon, David. “Output Routines: Examples and Techniques. Part III: Insertions”. *TUGboat* **11**(4), 588–605, 1990c.
- Schröder, Martin. *The everyshi package*, 2001. <http://mirrors.ctan.org/macros/latex/packages/ms>.
- Umeki, Hideo. *The geometry package*, 2010. <http://mirrors.ctan.org/macros/latex/contrib/geometry>.

- ◇ Boris Veytsman  
Computational Materials Science  
Center, MS 6A2  
George Mason University  
Fairfax, VA 22030  
[borisv \(at\) lk dot net](mailto:borisv@lk-dot-net)
- ◇ Michael Ware  
Department of Physics and  
Astronomy, N283 ESC  
Brigham Young University  
Provo, UT 84602  
[ware \(at\) byu dot edu](mailto:ware@byu-dot-edu)