

Perl and Excel

*Reading and Writing Excel for
Finance and Weather Applications*

Mike Schienle

Custom Visuals, LLC

mgs@customvisuals.com

<http://www.customvisuals.com>

Reading Excel Files

- ◆ Spreadsheet::ParseXLSX
 - ◆ Spreadsheet::XLSX listed as obsolete
 - ◆ \geq Excel 2007 Files (.xlsx)
- ◆ Spreadsheet::ParseExcel
 - ◆ \leq Excel 2003 Files (.xls)
- ◆ Interchangeable syntax

	A	B	C	D	E	F
1	Report	Source	Target	Start Time	Alert Time	Stop Time
2	AccrualPnL_PROD_MidnightSnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
3	AccrualPnL_PROD_TopDaySnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
4	FXRates_PROD_MidnightSnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
5	FXRates_PROD_TopDaySnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
6						

◆ Parsing

- ◆ \$xls is file object
- ◆ \$wb is workbook object
- ◆ \$ws is worksheet object(s)

	A	B	C	D	E	F
1	Report	Source	Target	Start Time	Alert Time	Stop Time
2	AccrualPnL_PROD_MidnightSnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
3	AccrualPnL_PROD_TopDaySnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
4	FXRates_PROD_MidnightSnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
5	FXRates_PROD_TopDaySnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
6						

◆ Workbooks and Worksheets

>= 2007

```
$xls = Spreadsheet::ParseXLSX->new()
    or $log->logconfess("Unable to create new XLSX Parser: ",
    $xls->error());
```

<= 2003 \$xls = Spreadsheet::ParseExcel->new() ...

```
$wb = $xls->parse($file)
    or $log->logconfess("Unable to parse Excel file: ",
    $xls->error());
```

```
$wsConfig = $wb->worksheet('Configuration')
    or $log->logconfess("Unable to access Config Sheet: ",
    $xls->error());
```


	A	B	C	D	E	F
1	Report	Source	Target	Start Time	Alert Time	Stop Time
2	AccrualPnL_PROD_MidnightSnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
3	AccrualPnL_PROD_TopDaySnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
4	FXRates_PROD_MidnightSnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
5	FXRates_PROD_TopDaySnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
6						

◆ Row and Column ranges

```

($rowMin, $rowMax) = $wsConfig->row_range();
($colMin, $colMax) = $wsConfig->col_range();
$log->info("Rows: $rowMin - $rowMax");
$log->info("Cols: $colMin - $colMax");

```


	A	B	C	D	E	F
1	Report	Source	Target	Start Time	Alert Time	Stop Time
2	AccrualPnL_PROD_MidnightSnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
3	AccrualPnL_PROD_TopDaySnap	Q:\FIRM\AccrualPnL\AccrualPnL_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\AccrualPNL_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
4	FXRates_PROD_MidnightSnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	8:30pm	9:15pm	11:59pm
5	FXRates_PROD_TopDaySnap	Q:\FIRM\FXRates\FXRates_YYYYMMDD_HHMMSS.csv	Q:\FTP\Out\FXRates_YYYYMMDD_HHMMSS.csv	4:00am	4:45am	10:15am
6						

◆ Rows, Columns and Cells

```

for $row ($rowMin .. $rowMax) {
    for $col ($colMin .. $colMax) {
        $cell = $wsConfig->get_cell($row, $col);
        if ($row == 0) {
            push @headers, $cell->value();
        }
        else {
            if ($headers[$col] =~ /Time/i) {
                $config{$row}->{$headers[$col]} =
                    ExcelFmt('hh:mm AM/PM', $cell->unformatted());
            }
            else {
                $config{$row}->{$headers[$col]} = $cell->value();
            }
        }
    }
}

```


Reading Formats and Fonts

- ◆ Useful for rebuilding a file with new input
- ◆ Copying formats between files
- ◆ I have little / no experience with reading them
- ◆ Commence hand-waving and non-answers
- ◆ Copy / pasted from CPAN docs

Reading Format Properties

◆ Formats

`$format->{Font}`

`$format->{AlignH}` (returns: 0 => None, 1 => Left, etc.)

`$format->{AlignV}` (returns: 0 => Top, 1 => Center, etc.)

`$format->{Indent}`

`$format->{Wrap}`

`$format->{Rotate}` (returns: 0 => None, 1 => Top down, etc.)

`$format->{JustLast}`

`$format->{ReadDir}`

`$format->{BdrStyle}`

`$format->{Fill}`

`$format->{Lock}`

`$format->{Hidden}`

`$format->{Style}`

Reading Font Properties

◆ Fonts

`$font->{Name}`

`$font->{Bold}`

`$font->{Italic}`

`$font->{Height}`

`$font->{Underline}`

`$font->{UnderlineStyle}` (returns: 0 => None, 1 => Single, etc.)

`$font->{Color}`

`$font->{Strikeout}`

`$font->{Super}` (returns: 0 => None, 1 => Superscript, etc.)

Writing Excel Files

- ◆ `Excel::Writer::XLSX`
 - ◆ `>=` Excel 2007 Files (.xlsx)
- ◆ `Spreadsheet::WriteExcel`
 - ◆ `<=` Excel 2003 Files (.xls)
- ◆ Interchangeable syntax

A1		fx Lon/Lat																						
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	Lon/Lat	30.0	30.5	31.0	31.5	32.0	32.5	33.0	33.5	34.0	34.5	35.0	35.5	36.0	36.5	37.0	37.5	38.0	38.5	39.0	39.5	40.0	40.5	41.0
2	-81.5	297.4	297.9	298.8	296.8	296.3	297.7	298.7	299.5	298.5	297	296.4	293.8	293.5	290.1	290.5	291.5	292.5	293.5	294.1	293.5	293.8	293.4	291.5
3	-81.0	300.4	300.8	301.1	301.1	299.1	297.2	298.7	299.7	299.6	298.3	297	296	293	291.1	290.7	291	291.1	292.8	293.1	293.3	293.4	292.6	291.2
4	-80.5	300.9	300.8	300.9	301.2	301.8	299.1	298.4	299.3	299.5	300	297.5	295.6	294.9	294.4	290.6	290.1	289.9	291.4	292.1	293	292.7	291.8	291.1
5	-80.0	301.3	301.2	301.1	301.1	301.4	301.5	298.4	298.8	299.8	300	297.3	294.5	295	294	292.5	291.1	289.3	289.2	291	292.4	292.7	291.9	290.9
6	-79.5	301.4	301.4	301.3	301.4	301.4	301.6	300.7	298.2	298.8	298.7	297.8	296	295.4	294.7	293.3	291.1	290	289.2	289.6	291.1	291.9	291.5	290.2
7	-79.0	301.1	301.1	301.1	301.4	301.6	301.5	301.6	300.5	298.3	297.8	297.4	296.7	295.9	295.1	294.2	292.9	292	290.7	290.6	290.6	290.8	291	289.5

✦ Writing

- ✦ \$wb is file / workbook object
- ✦ \$ws is worksheet object(s)
- ✦ Formats and Fonts have bigger role

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	Lon/Lat	30.0	30.5	31.0	31.5	32.0	32.5	33.0	33.5	34.0	34.5	35.0	35.5	36.0	36.5	37.0	37.5	38.0	38.5	39.0	39.5	40.0	40.5	41.0
2	-81.5	297.4	297.9	298.8	296.8	296.3	297.7	298.7	299.5	298.5	297	296.4	293.8	293.5	290.1	290.5	291.5	292.5	293.5	294.1	293.5	293.8	293.4	291.5
3	-81.0	300.4	300.8	301.1	301.1	299.1	297.2	298.7	299.7	299.6	298.3	297	296	293	291.1	290.7	291	291.1	292.8	293.1	293.3	293.4	292.6	291.2
4	-80.5	300.9	300.8	300.9	301.2	301.8	299.1	298.4	299.3	299.5	300	297.5	295.6	294.9	294.4	290.6	290.1	289.9	291.4	292.1	293	292.7	291.8	291.1
5	-80.0	301.3	301.2	301.1	301.1	301.4	301.5	298.4	298.8	299.8	300	297.3	294.5	295	294	292.5	291.1	289.3	289.2	291	292.4	292.7	291.9	290.9
6	-79.5	301.4	301.4	301.3	301.4	301.4	301.6	300.7	298.2	298.8	298.7	297.8	296	295.4	294.7	293.3	291.1	290	289.2	289.6	291.1	291.9	291.5	290.2
7	-79.0	301.1	301.1	301.1	301.4	301.6	301.5	301.6	300.5	298.3	297.8	297.4	296.7	295.9	295.1	294.2	292.9	292	290.7	290.6	290.6	290.8	291	289.5

◆ File/Workbook Properties

```

$wb = Excel::Writer::XLSX->new("$file");
$wb->set_properties(
    title      => 'GFS Weather Processor',
    author     => 'Mike Schienle <mgs@customvisuals.com>',
    keywords  => "GFS RunDate: $rDate, Forecast: $fDate"
);

$fmtPlain = $wb->add_format();
$fmtBold  = $wb->add_format(bold => 1);
$fmtGeo   = $wb->add_format(bold => 1, num_format => '0.0');

$ws = $wb->add_worksheet("Forecast $hour");

```


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	Lon/Lat	30.0	30.5	31.0	31.5	32.0	32.5	33.0	33.5	34.0	34.5	35.0	35.5	36.0	36.5	37.0	37.5	38.0	38.5	39.0	39.5	40.0	40.5	41.0
2	-81.5	297.4	297.9	298.8	296.8	296.3	297.7	298.7	299.5	298.5	297	296.4	293.8	293.5	290.1	290.5	291.5	292.5	293.5	294.1	293.5	293.8	293.4	291.5
3	-81.0	300.4	300.8	301.1	301.1	299.1	297.2	298.7	299.7	299.6	298.3	297	296	293	291.1	290.7	291	291.1	292.8	293.1	293.3	293.4	292.6	291.2
4	-80.5	300.9	300.8	300.9	301.2	301.8	299.1	298.4	299.3	299.5	300	297.5	295.6	294.9	294.4	290.6	290.1	289.9	291.4	292.1	293	292.7	291.8	291.1
5	-80.0	301.3	301.2	301.1	301.1	301.4	301.5	298.4	298.8	299.8	300	297.3	294.5	295	294	292.5	291.1	289.3	289.2	291	292.4	292.7	291.9	290.9
6	-79.5	301.4	301.4	301.3	301.4	301.4	301.6	300.7	298.2	298.8	298.7	297.8	296	295.4	294.7	293.3	291.1	290	289.2	289.6	291.1	291.9	291.5	290.2
7	-79.0	301.1	301.1	301.1	301.4	301.6	301.5	301.6	300.5	298.3	297.8	297.4	296.7	295.9	295.1	294.2	292.9	292	290.7	290.6	290.6	290.8	291	289.5

◆ Row / Column headers

set the lat and lon grid positions

```
$row = $col = $count = 0;
```

```
$ws->write_string($row, $col, 'Lon/Lat', $fmtBold);
```

```
for ($latPos = $dbData->{LatMin};
```

```
    $latPos <= $dbData->{LatMax};
```

```
    $latPos += $dataRes) {
```

```
    $mapLat{$latPos} = ++$count;
```

```
}
```

repeat for lonPos

```
for $latPos (keys %mapLat) {
```

```
    $ws->write_number($row, $mapLat{$latPos}, $latPos, $fmtGeo);
```

```
}
```

repeat for lonPos

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	Lon/Lat	30.0	30.5	31.0	31.5	32.0	32.5	33.0	33.5	34.0	34.5	35.0	35.5	36.0	36.5	37.0	37.5	38.0	38.5	39.0	39.5	40.0	40.5	41.0
2	-81.5	297.4	297.9	298.8	296.8	296.3	297.7	298.7	299.5	298.5	297	296.4	293.8	293.5	290.1	290.5	291.5	292.5	293.5	294.1	293.5	293.8	293.4	291.5
3	-81.0	300.4	300.8	301.1	301.1	299.1	297.2	298.7	299.7	299.6	298.3	297	296	293	291.1	290.7	291	291.1	292.8	293.1	293.3	293.4	292.6	291.2
4	-80.5	300.9	300.8	300.9	301.2	301.8	299.1	298.4	299.3	299.5	300	297.5	295.6	294.9	294.4	290.6	290.1	289.9	291.4	292.1	293	292.7	291.8	291.1
5	-80.0	301.3	301.2	301.1	301.1	301.4	301.5	298.4	298.8	299.8	300	297.3	294.5	295	294	292.5	291.1	289.3	289.2	291	292.4	292.7	291.9	290.9
6	-79.5	301.4	301.4	301.3	301.4	301.4	301.6	300.7	298.2	298.8	298.7	297.8	296	295.4	294.7	293.3	291.1	290	289.2	289.6	291.1	291.9	291.5	290.2
7	-79.0	301.1	301.1	301.1	301.4	301.6	301.5	301.6	300.5	298.3	297.8	297.4	296.7	295.9	295.1	294.2	292.9	292	290.7	290.6	290.6	290.8	291	289.5

◆ CSV file raw data:

RunDT, ForecastDT, Variable, Label, Longitude, Latitude, Value

"2015-06-18 00:00", "2015-06-18 00:00", "TMP", "surface", -81.5, 30, 297.4

"2015-06-18 00:00", "2015-06-18 00:00", "TMP", "surface", -81, 30, 300.4

◆ Read CSV file into array

```
@dataCSV = fileRead(file => $dbData->{FileCSV});
```

◆ Excel::Writer::XLSX code:

```
for $dataLine (@dataCSV) {
    ($lonPos, $latPos, $geoVal) = (split ",", $dataLine)[-3 .. -1];
    $ws->write_number($mapLon{$lonPos}, $mapLat{$latPos}, $geoVal);
}
```


Writing Excel Formulas

◆ Translate row / col to cell (xl_rowcol_to_cell)

```
while ($dbData = $sth->fetchrow_hashref()) {
    $row++;
    $col = 0;
    for (@fields) {
        $rcSt = xl_rowcol_to_cell(3, $col);
        $rcSp = xl_rowcol_to_cell($row, $col);
        if ($budgetField{$_} || $_ eq 'Total') {
            $ws->write_formula(1, $col, "=SUM($rcSt:$rcSp)");
        }
        else {
            $ws->write_formula(1, $col, "=COUNTA($rcSt:$rcSp)");
        }
        $col++;
    }
}
```


Writing Excel Formulas (obsolete)

- ◆ In Spreadsheet::WriteExcel it was computationally expensive to write formulas since they were parsed by a recursive descent parser. The `store_formula()` and `repeat_formula()` methods were used as a way of avoiding the overhead of repeated formulas by reusing a pre-parsed formula.
- ◆ In Excel::Writer::XLSX this is no longer necessary since it is just as quick to write a formula as it is to write a string or a number.

Writing Excel Formulas (obsolete)

```
# Orig: 2004/01/02 21:12:00

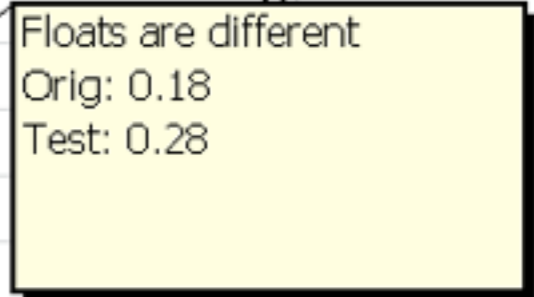
# store formulas for faster processing
$fDiff = $ws->store_formula('=IF(View!A100 <> "", view!A100,
    IF(Compare!A100 <> "", Compare!A100, ""))');

for ($rCount = 2; $rCount <= $maxRow; $rCount++) {
    for $c (qw/A B C D E F/) {
        $cr = $c . $rCount;
        $ws->repeat_formula($cr, $fDiff, undef,
            ('A100', "$cr") x 4);
    }
    $ws->write("G$rCount", "N/A");
}
```


Writing Excel Comments

```
$comment =  
    sprintf "%s are different\r\n" .  
    "Orig: %s\r\n" . "Test: %s\r\n",  
    $mapHeaders{dataType},  
    $recOrig{$MD5value}->{$header},  
    $recTest{$MD5value}->{$header};  
  
# adjust to scale  
$commentwidth = $commentheight = 0;  
for (split "\n", $comment) {  
    if (length($_) > $commentwidth) {  
        $commentwidth = length($_);  
    }  
    $commentheight++;  
}  
$ws->write_comment(  
    $row, $col, $comment,  
    x_scale => $commentwidth / 20,  
    y_scale => $commentheight / 4,  
);
```

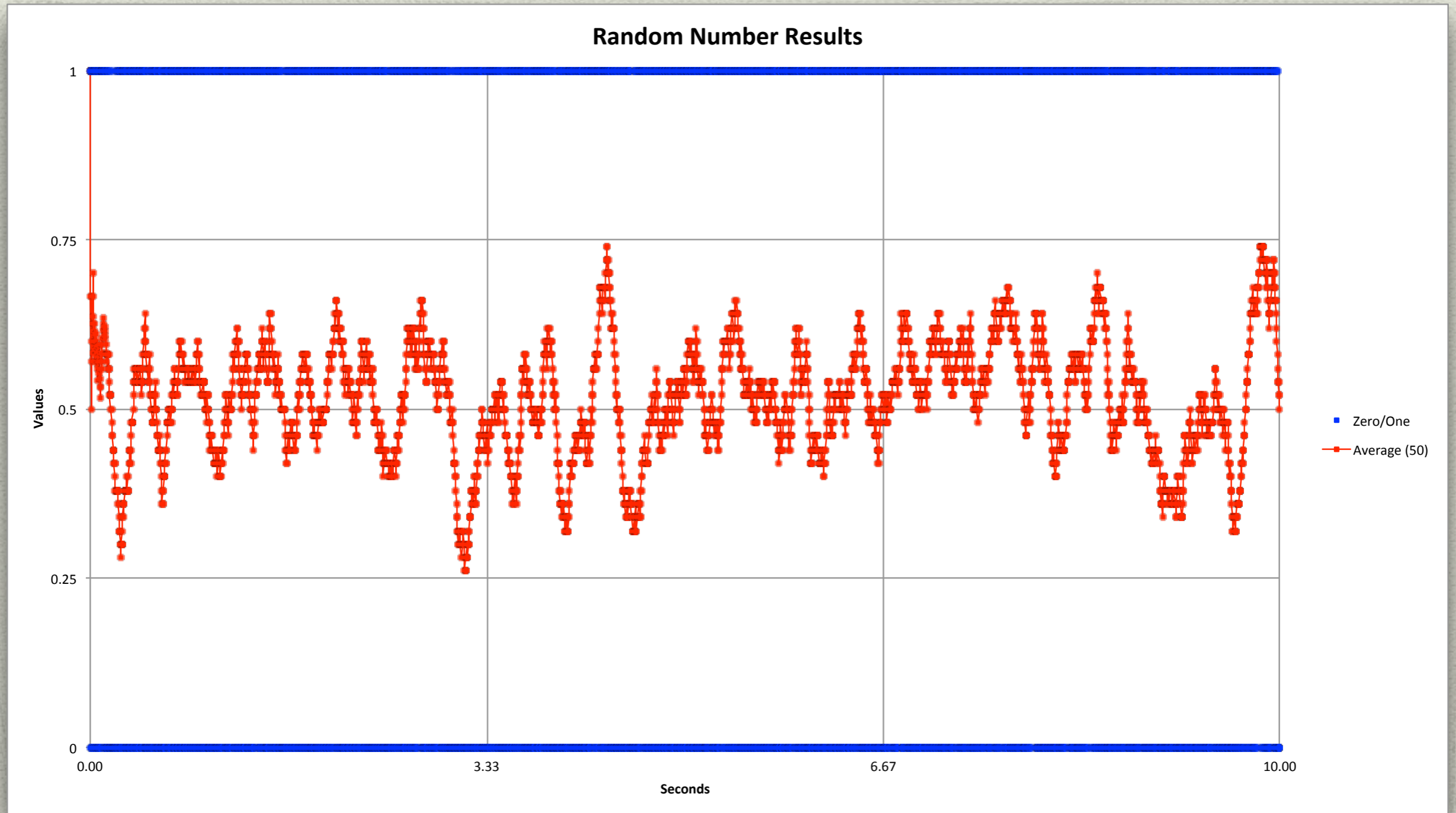
U	V	W
SD Qty	Current Face	Trade Factor
0	0	0
0	0	0
-0.1		0
0		0
0		0
0		0
0	0	0



Floats are different
Orig: 0.18
Test: 0.28

The image shows an Excel spreadsheet with three columns: U, V, and W. Column U is highlighted in yellow and contains the header 'SD Qty'. Column V is highlighted in light blue and contains the header 'Current Face'. Column W is highlighted in light blue and contains the header 'Trade Factor'. The data rows show values for 'SD Qty', 'Current Face', and 'Trade Factor'. A red box highlights the value '-0.1' in the 'SD Qty' column of the fourth row. A yellow comment box with a black border is overlaid on the spreadsheet, pointing to the '-0.1' value. The comment text reads: 'Floats are different', 'Orig: 0.18', and 'Test: 0.28'.

Writing Excel Charts



Writing Excel Charts

- ◆ Just another element of Spreadsheet
 - ◆ `add_chart` to workbook object
 - ◆ `$chart = $wb->add_chart(type => 'scatter');`
 - ◆ Set size, x/y/z axis params, grids, borders, data, markers, lines, colors, etc.


```

# Create a new Excel workbook
$wb = Excel::Writer::XLSX->new($file) or die "Unable to create $file: $!";

# Add a data worksheet
$data = $wb->add_worksheet('Random');

# write header row
$data->write($row, $col, ['Elapsed', 'Value', 'Average', 'Zeroes', 'Ones']);

# write data
for ($countMain = 0; $countMain <= $#data; $countMain++) {
    $startSub = ($countMain < 50) ? 0 : $countMain - 50;
    $sum = 0;
    for ($countSub = $startSub; $countSub <= $countMain; $countSub++) {
        $sum += $data[$countSub];
    }
    $avg = $sum / (($countMain < 50) ? $countMain + 1 : 50);
    $data->write($row, $col, [sprintf("%0.3f", $et[$countMain]),
        $data[$countMain], sprintf("%0.3f", $avg),
        $running{$countMain}->{0} || 0,
        $running{$countMain}->{1} || 0], $fmt);
    $row++;
}

$data->autofilter(0, 0, $#data + 1, 4);
$data->freeze_panes(1, 0);

```



```

# Add a chart
$chart = $wb->add_chart(type => 'scatter', embedded => 1);

# set a decent size
$chart->set_size(width => 1200, height => 800);

# X axis
$chart->set_x_axis(
    name           => 'Seconds',
    min            => 0,
    max            => $time,
    minor_unit     => $time / 12,
    major_unit     => $time / 3,
    major_gridlines => {visible => 1},
    num_format     => '0.00',
);

# Y axis
$chart->set_y_axis(
    name           => 'values',
    min            => 0,
    max            => 1,
    minor_unit     => 0.125,
    major_unit     => 0.25,
    major_gridlines => {visible => 1},
);

```



```

# Zeroes/Ones
$chart->add_series(
  name      => 'Zero/One',
  categories => ['Random', 1, $#data + 1, 0, 0],
  values    => ['Random', 1, $#data + 1, 1, 1],
  marker    => {
    type     => 'square',
    size     => 3,
    border   => {color => 'blue'},
    fill     => {color => 'blue'},
  },
);

```

```

# Running Averages
$chart->add_series(
  name      => 'Average (50)',
  categories => ['Random', 1, $#data + 1, 0, 0],
  values    => ['Random', 1, $#data + 1, 2, 2],
  line      => {
    color    => 'red',
    width    => 1
  },
  marker    => {
    type     => 'square',
    size     => 3,
    border   => {color => 'red'},
    fill     => {color => 'red'},
  },
);

```



```
# chart title and axis labels
$chart->set_title(name => 'Random Number Results');

$data->insert_chart(1, 5, $chart, 10, 10);

$wb->close() or die "Unable to close $file: $!";
```

