Introduction to the SCAN.UPC Package

Stephen R. Piccolo

October 17, 2016

Contents

1 Background

This vignette describes how to normalize samples with the *Single-Channel Array Normalization* (SCAN) and *Universal exPression Codes* (UPC) methods. The motivations and methodology behind these approaches have been described in detail in recent papers (**??**). Briefly, SCAN and UPC are quantitative approaches for normalizing gene-expression data. The SCAN method produces standardized expression estimates for one-color and two-color microarrays and is most suitable for analyses where a single such technology is used. The UPC method produces standardized expression values that estimate whether a given gene (or transcript or exon) is "active" in a given sample. UPCs can be generated for microarrays or RNA-Seq data and can be interpreted consistently, irrespective of the technology/platform used for profiling; thus UPCs are suitable for integrating expression data across multiple platforms.

A distinguishing feature of the SCAN and UPC methods is that they are applied to *individual* samples. This means that the output for a given sample will be the same whether the sample is processed in isolation or jointly with other samples. This feature also has computational advantages: when a large batch of samples needs to be processed, it is not necessary to store the entire batch in computer memory at the same time.

SCAN and UPC correct for technological and experimental biases that can arise during expression profiling. For example, microarray probes with high G/C content tend to be biased toward higher overall expression. And in RNA-Seq experiments, the G/C content, size of genomic regions being profiled, and read depth can lead to biases. The SCAN and UPC algorithms correct for such factors and standardize variances across probes/regions. A two-component mixture model estimates which probes/regions constitute background noise or biological signal.

2 How to produce SCAN estimates for Affymetrix microarrays

SCAN can be applied to any Affymetrix microarray for which an annotation package (constructed via the pdInfoBuilder package) exists in Bioconductor. This section demonstrates how to normalize a raw Affymetrix microarray file. In the examples below, a CEL file is downloaded from Gene Expression Omnibus (GEO), saved to a temporary file, and normalized using SCAN. Various optional parameters are also demonstrated.

Initially, we will download a CEL file via the GEOquery package.

```
> library(GEOquery)
> tmpDir = tempdir()
> library(GEOquery)
> getGEOSuppFiles("GSM555237", makeDirectory=FALSE, baseDir=tmpDir)
> celFilePath = file.path(tmpDir, "GSM555237.CEL.gz")
```

To normalize the sample, we invoke the `SCAN` function. This function requires one mandatory parameter: a path specifying the location of the file to be normalized.

```
> library(SCAN.UPC)
> normalized = SCAN(celFilePath)
```

For convenience, it is also possible to download microarray samples from GEO and normalize them in a single step. To do this, substitute the file path with a GEO identifier.

```
> normalized = SCAN("GSM555237")
```

The `SCAN` function returns an *ExpressionSet* object containing a row for each probeset value. Detailed status information, including the number of iterations required for mathematical convergence of the mixture model, is printed to the console.

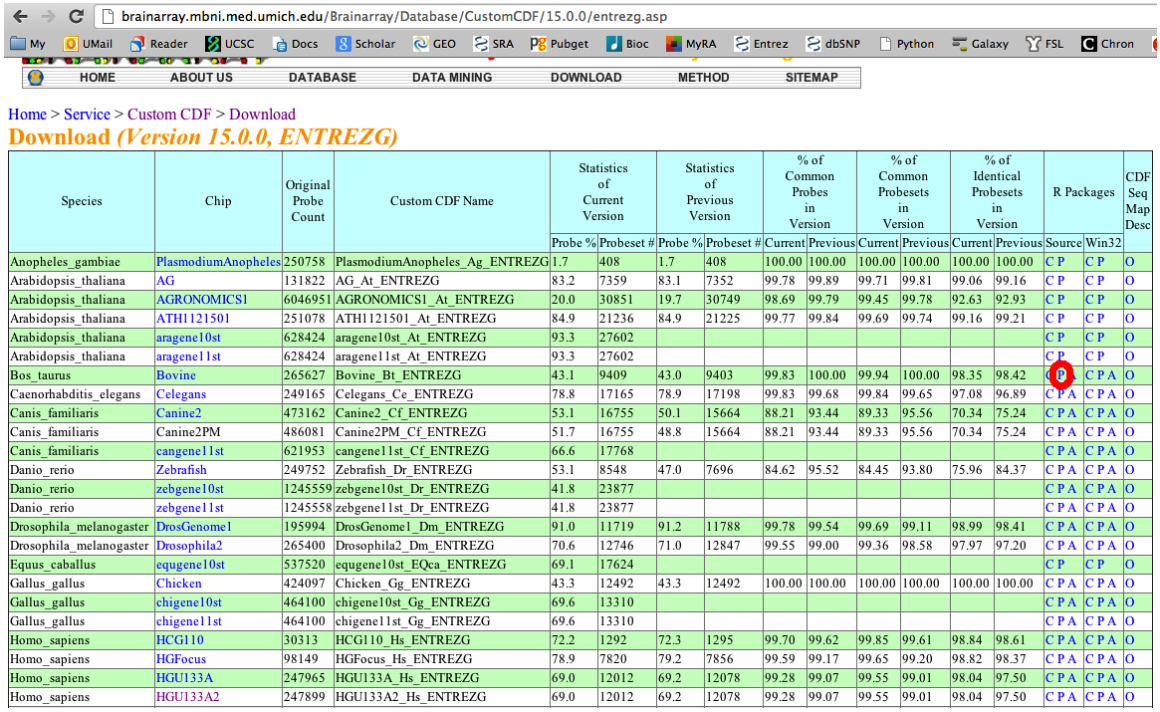
Multiple input files can be processed in one command via specifying wildcard characters (e.g., "*.CEL") or GEO experiment identifiers (e.g., "GSE22309"). In this case, the `SCAN` function returns an *ExpressionSet* object with a row for each probeset and a column for each input file.

Using the optional `outFilePath` parameter, the normalized values can be saved to a text file. The example below demonstrates this option.

```
> normalized = SCAN(celFilePath, outFilePath="output_file.txt")
```

By default, `SCAN` maps Affymetrix probes to "probeset" identifiers provided by the manufacturer. However, these mappings may be outdated and include problematic probes (for example, those that may cross hybridize). In addition, multiple probesets may be assigned to a single gene. As an alternative, the BrainArray resource (see `http://brainarray.mbni.med.umich.edu/Brainarray/Database/CustomCDF/genomic_curated_CDF.asp`) provides regularly updated mappings that exclude problematic probes and map to genes rather than probesets. When invoking `SCAN`, users can specify such alternative mappings via the `probeSummaryPackage` parameter. (Packages other than those provided by BrainArray can be used if they conform to the standards of the AnnotationDbi package.)

BrainArray packages can be downloaded manually from `http://brainarray.mbni.med.umich.edu/Brainarray/Database/CustomCDF/CDF_download.asp`. When downloading, be sure to download the R source package for probe-level mappings (example below).



brainarray.mbni.med.umich.edu/Brainarray/Database/CustomCDF/15.0.0/entrezg.asp

My UMail Reader UCSC Docs Scholar GEO SRA Pubget Bioc MyRA Entrez dbSNP Python Galaxy FSL Chron

HOME ABOUT US DATABASE DATA MINING DOWNLOAD METHOD SITEMAP

Home > Service > Custom CDF > Download

Download *(Version 15.0.0, ENTREZG)*

Species	Chip	Original Probe Count	Custom CDF Name	Statistics of Current Version		Statistics of Previous Version		% of Common Probes in Version		% of Common Probesets in Version		% of Identical Probesets in Version		R Packages		CDF Seq Map Desc
				Probe %	Probeset #	Probe %	Probeset #	Current	Previous	Current	Previous	Current	Previous	Source	Win32	
Anopheles_gambiae	PlasmodiumAnopheles	250758	PlasmodiumAnopheles_Ag_ENTREZG	1.7	408	1.7	408	100.00	100.00	100.00	100.00	100.00	100.00	C P	C P	O
Arabidopsis_thaliana	AG	131822	AG_At_ENTREZG	83.2	7359	83.1	7352	99.78	99.89	99.71	99.81	99.06	99.16	C P	C P	O
Arabidopsis_thaliana	AGRONOMICS1	6046951	AGRONOMICS1_At_ENTREZG	20.0	30851	19.7	30749	98.69	99.79	99.45	99.78	92.63	92.93	C P	C P	O
Arabidopsis_thaliana	ATH1121501	251078	ATH1121501_At_ENTREZG	84.9	21236	84.9	21225	99.77	99.84	99.69	99.74	99.16	99.21	C P	C P	O
Arabidopsis_thaliana	aragene10st	628424	aragene10st_At_ENTREZG	93.3	27602									C P	C P	O
Arabidopsis_thaliana	aragene11st	628424	aragene11st_At_ENTREZG	93.3	27602									C P	C P	O
Bos_taurus	Bovine	265627	Bovine_Bt_ENTREZG	43.1	9409	43.0	9403	99.83	100.00	99.94	100.00	98.35	98.42	C P	C P A	O
Caenorhabditis_elegans	Celegans	249165	Celegans_Ce_ENTREZG	78.8	17165	78.9	17198	99.83	99.68	99.84	99.65	97.08	96.89	C P A	C P A	O
Canis_familiaris	Canine2	473162	Canine2_Cf_ENTREZG	53.1	16755	50.1	15664	88.21	93.44	89.33	95.56	70.34	75.24	C P A	C P A	O
Canis_familiaris	Canine2PM	486081	Canine2PM_Cf_ENTREZG	51.7	16755	48.8	15664	88.21	93.44	89.33	95.56	70.34	75.24	C P A	C P A	O
Canis_familiaris	cangene11st	621953	cangene11st_Cf_ENTREZG	66.6	17768									C P A	C P A	O
Danio_rerio	Zebrafish	249752	Zebrafish_Dr_ENTREZG	53.1	8548	47.0	7696	84.62	95.52	84.45	93.80	75.96	84.37	C P A	C P A	O
Danio_rerio	zebgene10st	1245559	zebgene10st_Dr_ENTREZG	41.8	23877									C P A	C P A	O
Danio_rerio	zebgene11st	1245558	zebgene11st_Dr_ENTREZG	41.8	23877									C P A	C P A	O
Drosophila_melanogaster	DrosGenome1	195994	DrosGenome1_Dm_ENTREZG	91.0	11719	91.2	11788	99.78	99.54	99.69	99.11	98.99	98.41	C P A	C P A	O
Drosophila_melanogaster	Drosophila2	265400	Drosophila2_Dm_ENTREZG	70.6	12746	71.0	12847	99.55	99.00	99.36	98.58	97.97	97.20	C P A	C P A	O
Equus_caballus	equgene10st	537520	equgene10st_EQca_ENTREZG	69.1	17624									C P	C P	O
Gallus_gallus	Chicken	424097	Chicken_Gg_ENTREZG	43.3	12492	43.3	12492	100.00	100.00	100.00	100.00	100.00	100.00	C P A	C P A	O
Gallus_gallus	chigene10st	464100	chigene10st_Gg_ENTREZG	69.6	13310									C P A	C P A	O
Gallus_gallus	chigene11st	464100	chigene11st_Gg_ENTREZG	69.6	13310									C P A	C P A	O
Homo_sapiens	HCG110	30313	HCG110_Hs_ENTREZG	72.2	1292	72.3	1295	99.70	99.62	99.85	99.61	98.84	98.61	C P A	C P A	O
Homo_sapiens	HGFocus	98149	HGFocus_Hs_ENTREZG	78.9	7820	79.2	7856	99.59	99.17	99.65	99.20	98.82	98.37	C P A	C P A	O
Homo_sapiens	HGU133A	247965	HGU133A_Hs_ENTREZG	69.0	12012	69.2	12078	99.28	99.07	99.55	99.01	98.04	97.50	C P A	C P A	O
Homo_sapiens	HGU133A2	247899	HGU133A2_Hs_ENTREZG	69.0	12012	69.2	12078	99.28	99.07	99.55	99.01	98.04	97.50	C P A	C P A	O

After such a package has been downloaded, it can be installed using code such as the following.

```
> install.packages("hgu95ahsentrezgprobe_15.0.0.tar.gz", repos=NULL, type="source")
```

Or instead of downloading BrainArray packages manually, it is now possible to download these packages via the `InstallBrainArrayPackage` function.

```
> pkgName = InstallBrainArrayPackage(celFilePath, "15.0.0", "hs", "entrezg")
```

These mappings can be applied during normalization using code such as the following.

```
> normalized = SCAN(celFilePath, probeSummaryPackage=pkgName)
```

It is also possible to adjust the data set for batch effects. Please see the description of the `batchFilePath` parameter in the documentation.

3 How to produce SCAN estimates for Agilent two-color microarrays

The SCAN.UPC package also supports the ability to normalize Agilent two-color microarrays. The general concept is similar to Affymetrix arrays; however, SCAN also corrects for biases that can arise due to the dyes used in each channel, as well as inter-channel correlation. (This package does not yet support normalizing Agilent one-color arrays.)

To normalize Agilent two-color arrays, use the `SCAN_TwoColor` function. In the example below, an example file is downloaded from GEO and then normalized and saved to an output file.

```
> SCAN_TwoColor("GSM1072833", "output_file.txt")
```

4 How to produce UPC estimates for microarrays or RNA-Seq

The SCAN algorithm uses a two-component mixture model to distinguish expression levels that constitute background noise from those that represent biological signal. This model can be used to estimate whether a given gene is active (i.e., belongs to the mixture component representing biological signal). These Universal exPression Codes (UPCs) range between zero and one: a high value suggests a greater likelihood that a given gene is active in the sample, whereas a low value indicates the opposite. This package contains functions for applying this methodology to Affymetrix microarrays, Agilent two-color microarrays, and RNA-Seq read counts.

The SCAN.UPC package contains a series of functions that can be used to derive UPCs. These functions have similar parameters to their SCAN counterparts. As shown in the examples below, the `UPC` function produces UPC values for Affymetrix microarrays, and the `UPC_TwoColor` function can be applied to Agilent data.

```
> upc1 = UPC("GSM555237")
> upc2 = UPC_TwoColor("GSM1072833")
```

UPC values also can be derived for RNA-Seq read counts via the `UPC_RNASeq` function. These read counts might be generated via the commonly used Tophat short-read aligner (`http://tophat.cbcb.umd.edu/`) followed by summarization via the htseq-count tool (`http://www-huber.embl.de/users/anders/HTSeq/`); however, alternative tools could also be used.

The `UPC_RNASeq` function requires an input file that specifies a read count for each genomic region (e.g., gene). This file should list a unique identifier for each region in the first column and corresponding read counts (not RPKM/FPKM values) in the second column.

AAB	4486
AAC	10
AAD	0
AAE	88223

`UPC_RNASeq` can correct for the GC content and length of each genomic region. Users who wish to perform this correction must provide an annotation file. This tab-separated file should contain a row for each genomic region. The first column should contain a unique identifier that corresponds to identifiers from the read-count input file. The second column should indicate the length of the genomic region. And the third column should specify the number of G or C bases in the region.

AAB	1767	640
AAC	654	333
AAD	4644	2039
AAE	2629	1011

Alternatively, annotation files can be generated via the `ParseMetaFromGtfFile` function. This function parses gene length and GC information from GTF files (see `http://uswest.ensembl.org/info/website/upload/gff.html`), which are used commonly for RNA-Seq analyses.

After an annotation file has been generated, RNA-Seq read counts can be processed using code such as the following.

```
> upc3 = UPC_RNASeq("ReadCounts.txt", "Annotation.txt")
```

Note: In the current version of this package, it is now possible to UPC normalize data from any expression-profiling platform. For example, a user can now download any preprocessed data set from GEO and UPC transform the data in one line of code. The documentation describes the `UPC_Generic_ExpressionSet` and `UPC_Generic` functions, which enable this functionality.

5 Reducing processing time for Affymetrix microarray normalization

Because the SCAN algorithm accounts for nucleotide-level genomic composition across thousands of probes, it may take several minutes to normalize a sample, depending on the computer's processor speed and the type of microarray. To enable users to normalize samples in a shorter period of time, we have provided an alternative function called `SCANfast` (and its UPC counterpart, `UPCfast`). In this approach, a smaller number of probes is used for normalization, and a less stringent convergence threshold is used. We have found that microarrays processed with `SCANfast` (using default parameters) require 75% less processing time (on average) but produce output values that correlate strongly ($r = 0.998$) with values produced by the `SCAN` function for the same arrays. Parameter options for `SCANfast` are identical to those for `SCAN`.

In addition, we have made it possible to execute `SCAN`, `UPC`, `SCANfast`, or `UPCfast` in parallel. This approach uses the foreach package behind the scenes. If you have registered a parallel backend (for example, via the doParallel package), multiple CEL files can be processed in parallel. Otherwise, the files will be processed sequentially. The example below demonstrates how to normalize multiple files in parallel on multiple cores within a given computer. However, it is also possible using the doParallel package to spread the workload across multiple computers on a cluster.

```
> library(doParallel)
> registerDoParallel(cores=2)
> result = SCAN("GSE22309")
```

6 Conclusion

Please see the SCAN.UPC documentation for full descriptions of functions and the various options they support.