Calculating absolute and relative protein abundance from mass spectrometry-based protein expression data

Christine Vogel & Edward M Marcotte

Center for Systems and Synthetic Biology, Institute for Cellular and Molecular Biology, University of Texas at Austin, 2500 Speedway, MBB 3.210, Austin, Texas 78712, USA. Correspondence should be addressed to C.V. (cvogel@mail.utexas.edu) or E.M.M. (marcotte@icmb.utexas.edu).

Published online 21 August 2008; doi:10.1038/nprot.2008.132

Mass spectrometry (MS)-based shotgun proteomics allows protein identifications even in complex biological samples. Protein abundances can then be estimated from the counts of tandem MS (MS/MS) spectra attributable to each protein, provided one accounts for differential MS detectability of contributing peptides. We developed a method, APEX, which calculates Absolute Protein EXpression levels based upon learned correction factors, MS/MS spectral counts and each protein's probability of correct identification. This protocol describes APEX-based calculations in three parts. (i) Using training data, peptide sequences and their sequence properties, a model is built to estimate MS detectability (O_i) for any given protein. (ii) Absolute protein abundances are calculated from spectral counts, identification probabilities and the learned O_i-values. (iii) Simple statistics allow calculation of differential expression in two distinct biological samples, i.e., measuring relative protein abundances. APEX-based protein abundances span 3–4 orders of magnitude and are applicable to mixtures of 100s to 1,000s of proteins.

INTRODUCTION

Mass spectrometry (MS)-based shotgun proteomics is a fast and relatively easy method for protein identification. A shotgun proteomics experiment typically proceeds by tandem MS (MS/MS) analysis of peptides from proteolytically digested proteins, followed by *in silico* matching of the MS/MS spectra against a database of theoretical peptide spectra derived from protein sequences. Proteins are identified from combined evidence for their composite peptides, resulting in a list in which each protein is associated with a confidence score of correct identification, e.g., from Protein Prophet[1]. In addition, an MS dataset provides information on the types and number of peptide spectra associated with each protein, as well as peak heights.

A number of approaches have been developed to quantify protein observations from peak heights in shotgun proteomics experiments by introducing internal reference standards (e.g., ref. 2), often by addition of isotopically labeled peptides[3,4]. These reference standards can be derived from cells grown in labeled media, as in SILAC (stable isotope labeling with amino acids in cell culture)[5], by derivatizing natural samples, as in ICAT (isotope-coded affinity tags)[6], or can instead be synthesized and added to samples, as in isotope dilution (e.g., AQUA[7]). The necessity (and expense) of synthesizing thousands of isotopically labeled peptides has prevented easy scaling to full proteomes, even when employing unlabeled peptides[8].

Thus, development of label-free quantitation methods for MS has been of high interest. Recently, approaches have considered quantitation from the MS/MS sampling statistics in a shotgun proteomics experiment. Both the coverage of unique peptides in a protein (i.e., percentage of possible peptides per protein actually observed) and the total number of repeat observations of MS/MS spectra from all peptides in a protein (spectral count) approximate protein abundance[9–16]. However, both measures have shortcomings, such as coverage showing saturation (at 100%), spectral counts not accounting for protein size (larger proteins contribute more peptides), both approaches ignoring sampling depth, and neither approach considering the prior odds of observing any particular peptide in the experiment. Peptides vary considerably in their ability to be detected by an MS instrument due to, for example, chemical sequence properties that affect peptide ionization[17]. Although such trends can be partly predicted from a peptide's amino acid composition[18–24], previous quantitation approaches have not incorporated these predictions to adjust observed spectral counts.

We developed a method, called APEX (Absolute Protein EXpression index), which uses protein identification scores, spectral counts and prior estimates of the number of unique tryptic peptides expected for the protein (O_i-value) to calculate absolute protein expression indices[25]. We estimate the O_i-value employing machine learning techniques accounting for protein size, sequence properties, ionizability and other properties influencing MS detectability. The number of MS/MS spectra observed in the experiment, i.e., repeat peptide observations, is then normalized by the O_i-value for each protein, i.e., the number of unique peptides expected, and serves as an estimate of the protein's abundance. We also normalize by the total number of spectra observed in the experiment to enable comparison between experiments with different sampling depths.

APEX is a robust and rapid method to quantify absolute protein abundance. It is appropriate for large-scale protein expression measurements where absolute abundance estimates are desirable and especially where isotope labeling is impractical. In contrast to other non-MS-based techniques[26–29], APEX is simple to use for large-scale data sets and differential protein expression and does not require construction of fusion protein libraries, labeling or internal standards.

APEX-based protein abundances span 3–4 orders of magnitude and are applicable to mixtures of 100s to 1,000s of proteins[25]. We developed and tested APEX on two different electrospray ionization MS instruments (ThermoFinnigan Surveyor/DecaXP+ iontrap

(LCQ), ThermoFinnigan LTQ-OrbiTrap); however, the method is equally applicable to other MS instruments. APEX has been successfully applied to proteomes of yeast[25,30], *Escherichia coli*, mouse[25], *Mycobacterium*[31], and *Arabidopsis*[32], as well as human (C.V., E.M.M., L.O. Penalva, unpublished data). Related methods based on spectral counting were used, for example, for the fission yeast proteome[33].

This protocol describes APEX in three sections (**Fig. 1**). First, using a high-quality MS dataset, vectors of sequence features and machine learning techniques, we build a model to predict peptide MS detectability (Section 1A in **Fig. 1**). The resulting model is organism- and sequence-independent and can be reused for any set of sequences analyzed on the same MS instrument; the training and testing section can be omitted in further experiments. Then, using the model and amino acid sequence features, we predict protein MS detectability (O_i-values) as the sum of the respective peptide detectabilities (Section 1B in **Fig. 1**). This section is very similar to Section 1A with respect to preparation of the input data files. However, peptide observations are not known but predicted using the model created in Section 1A. Once O_i-values have been calculated for a particular set of sequences and experimental setup, this step can be omitted in future analyses.

Second, using postprocessed MS data, O_i-values for the detected proteins and an estimate of the total number of molecules per cell (C), we calculate indices of APEX for a given protein i (Section 2 in **Fig. 1**).

Third, for detection of relative protein abundances in two different samples, we present a test for statistically significant differential protein expression (Section 3 in **Fig. 1**). The statistical test (Z-score) is based only on spectral counts; for an estimate of expression fold change between the two samples APEX expression values need to be calculated as described in Steps 20–23.

We describe this protocol with the example of yeast cell lysate analyzed on the LTQ-OrbiTrap. At http://www.marcottelab.org/APEX_Protocol/, we provide input and output files created during the process, a suite of corresponding Perl scripts as well as data from analysis on the LCQ. We also provide example data for training and prediction of MS detectability of *E. coli*, yeast and human proteins both for the LTQ-OrbiTrap and the LCQ, as well as a Z-score analysis of yeast grown in minimal and rich medium. The models trained on these (or other) datasets can analyze data of any origin if the same parameters have been used for data postprocessing.

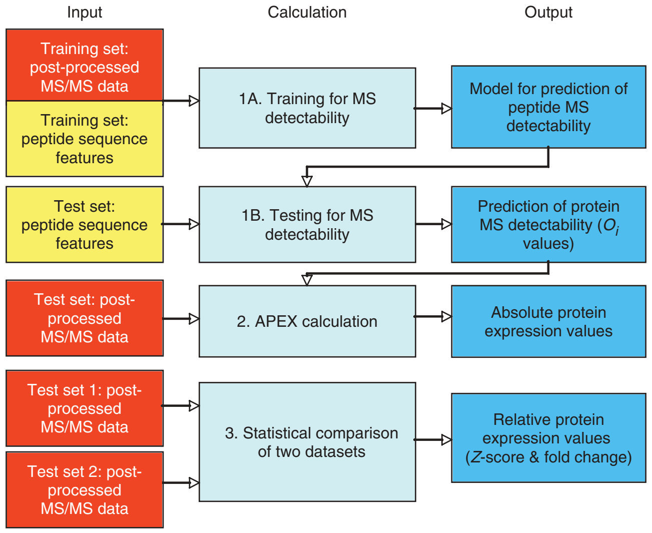

Input
Calculation
Output
Training set: post-processed MS/MS data
Training set: peptide sequence features
1A. Training for MS detectability
Model for prediction of peptide MS detectability
Test set: peptide sequence features
1B. Testing for MS detectability
Prediction of protein MS detectability (O_i values)
Test set: post-processed MS/MS data
2. APEX calculation
Absolute protein expression values
Test set 1: post-processed MS/MS data
Test set 2: post-processed MS/MS data
3. Statistical comparison of two datasets
Relative protein expression values (Z-score & fold change)

Figure 1 | APEX pipeline—overview. The protocol describes three different calculations. (1A) Using training MS/MS data, a model is created to describe peptide MS detectability. (1B) This model is then used to predict peptide MS detectability for any test data file. (2) Using O_i-values (summed probabilities of peptide MS detectability) and MS/MS data, we calculate APEX, an estimate of absolute protein expression. (3) Two MS/MS data files can be statistically compared calculating a Z-score. Fold-changes of expression levels are based on APEX estimates.

MATERIALS

EQUIPMENT

- MS raw or postprocessed data of proteolytic peptides from complex protein mixture
- Mac, PC or Linux/Unix workstation
- Amino acid sequences of proteins of interest, e.g., FASTA file
- Information on amino acid properties, e.g., aaindex1 file from ftp://ftp.genome.jp/pub/db/community/aaindex/
- Software to analyze MS raw data (Sequest, Mascot; Peptide Prophet[34] and ProteinProphet[1], see http://tools.proteomecenter.org/TPP.php)
- Scripting language for text parsing (e.g., Perl, Python). For a collection of sample scripts, see http://www.marcottelab.org/APEX_Protocol/
- WEKA (http://www.cs.waikato.ac.nz/~ml/weka/) machine learning software

PROCEDURE

Training and testing of a model for prediction of peptide and protein MS detectability: training

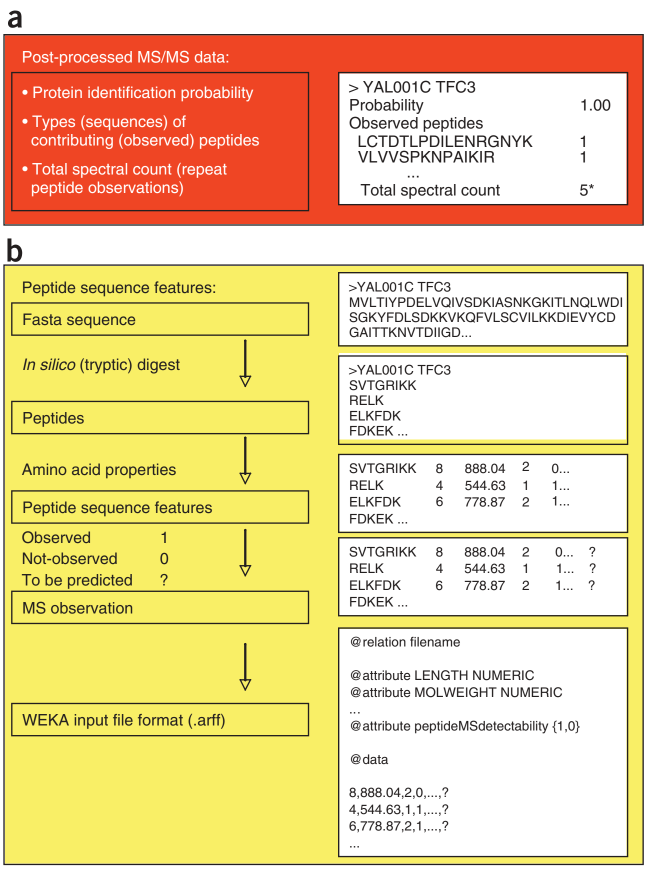
▲ CRITICAL STEP Steps 1–11 (training) can be omitted if a model has been built and saved in previous calculations for a particular MS instrument and setup. We found empirically that models are similar between MS instruments using the same ionization method and mass range, and the resulting O_i-values correlate. However, since, for example, an LCQ is less sensitive than an LTQ-OrbiTrap, O_i-values are generally smaller on the former instrument than on the latter (see **Supplementary Note** online).

1| Postprocess MS/MS raw data using software of choice (Sequest, Mascot, PeptideProphet[34] and ProteinProphet[1]) and parse for proteins of confident identification (e.g., false discovery rate $<5\%$).

2| From these proteins, select a set of ~30–150 proteins identified at high confidence. Even for these well-identified proteins, not all theoretically possible peptides will be observed, and the observation/nonobservation of peptides mapping to the proteins is used for training (**Fig. 2a**).

▲ CRITICAL STEP Selection of high-quality training data is crucial for successful model building and model performance. The training set of proteins (and its size) should be chosen so that (i) recall and precision (F-measure) in cross-validation are maximized

Figure 2 | Preparation of input files. We use two basic types of input data. (**a**) Postprocessed MS/MS data from which information on the probability of correct protein identification (*pi*), the types of contributing (observed) peptides and the number of their MS/MS spectral observations is extracted. A total of five MS/MS spectra map to the example protein, YAL001C. (**b**) Sequence feature data calculated for *in silico* digested protein sequences using known amino acid features. The feature vectors can be extended to any length; the most important features are described in literature[23,25]. The example protein YAL001C is described for a prediction of peptide MS detectability; for its peptides the panels list the length, molecular weight and two arbitrary features. *Total spectral count per protein.



(see Step 10) and (ii) time for calculation of the model is within desired time frame. In general, the larger the fraction of *observed* versus *nonobserved* peptides in the dataset, the better is model performance. This fraction seems more important than the actual number of proteins (or peptides) contained in the training dataset (see **Supplementary Note** online for details). However, the larger the training dataset, the more time is required to build a model.

We tested selection of the training dataset based on high protein and/or peptide identification probabilities as well as high-spectral counts per protein/per peptide. Alternatively, training proteins could also be chosen based on their presence in other experiments (e.g., in western blot data). We obtained better models when filtering for high-protein identification probability (e.g., 1.00) and high-spectral counts per protein (e.g., >200) than when filtering for high probabilities/spectral counts per peptide. However, note that these cutoffs are MS/MS dataset- and machine-dependent and should be reevaluated for different experimental setups. Our cutoffs provide a guideline for experimentation; the **Supplementary Note** online contains more detailed information.

For example, when creating a training file for a ThermoFinnigan LTQ-OrbiTrap, we analyzed yeast cellular lysate identifying 89 proteins of high-protein identification probability ($p_i = 1.00$) and with at least 200 total spectral counts per protein. For these proteins, 9% (1,331) of the peptides were observed in the MS/MS experiment; 91% (13,279) of peptides were not observed.

3| Digest the amino acid sequences for the proteins *in silico* into (tryptic) peptides, for example using Proteogest[35] at http://www.utoronto.ca/emililab/proteogest.htm/ Trypsin cleaves after Lys (K) or Arg (R) unless they are followed by Pro (P) (**Fig. 2b**). *In silico* digestions usually account for one or two missed cleavages per peptide which strongly increase the number and types of peptides per protein. In our example, we always include up to two missed cleavages. If only one or zero missed cleavages are allowed, the model has to be rebuilt. For model building, it is sufficient to digest only the proteins in the training dataset; however, we typically digest the whole proteome and then select the respective training proteins (see http://www.marcottelab.org/APEX_Protocol/ for Perl scripts). The choice of the maximum allowed number of missed tryptic cleavages should be the same for training, testing and application of APEX.

4| Describe sequence features (attributes) for all peptides. Attributes should include the peptide length (number of amino acids) and the amino acid frequencies (relative and absolute). Attributes can also include the molecular weight, number of unique theoretical peptides, hydrophobicity, solubility, solvent accessibility, etc. or features identified by Mallick *et al.*[23] to characterize proteotypic peptides. We collected all amino acid features from the AAindex (http://www.genome.jp/aaindex/). Attributes can be numerical, continuous or categorical. Consistent with Mallick *et al.*'s work, we include both the *sum* and the *average* values for any amino acid characteristic as a peptide feature.
▲ CRITICAL STEP The number and types of attributes included is important for model performance (see **Supplementary Note** online and http://www.marcottelab.org/APEX_Protocol/). When testing the same training dataset but with different sets of peptide attributes: (i) two attributes (length, molecular weight); (ii) 22 attributes (length, molecular weight and relative amino acid frequencies); (iii) 58 attributes (length, molecular weight, relative and absolute amino acid frequencies, secondary structure, five attributes identified by Mallick *et al.*[23]); and (iv) 66 attributes (as in (iii) plus four additional attributes).

We observed improving performance from (i) to (iv), with the largest improvement between (i) and (ii). For both the final model for the LCQ and the LTQ-OrbiTrap we use files with 66 peptide sequence attributes (iv). The attributes are listed in the **Supplementary Note** online.

5| For each of the peptides assign '1' if it has been observed in the selected proteomics data (Step 2) and '0' if it has not been observed. When using Peptide- and ProteinProphet output, observation of a peptide is marked as 'Contributing_peptide="Y"' in the –prot.xml file.

6| Convert the peptide feature vectors including MS observation (1,0) into WEKA .arff file format (**Fig. 2b**, **Supplementary Note** online) which lists all features (attributes) in the order in which they occur in the feature vector, as well as the feature vectors in form of comma separated values. The file format does not contain peptide identifiers; they need to be stored separately.

7| Create a model of peptide MS detectability using WEKA. The process requires a lot of computer memory (depending on the size of the training set), thus we recommend allocating extra memory to WEKA when opening it or using the command line options. Here, we describe the steps to be taken with WEKA Explorer Java user interface. To open WEKA and allocate 500 MB memory, enter 'java -Xmx512m -jar <your directory here>/weka.jar'. Computing times quoted here are obtained allocating 1,800 MB of memory to WEKA with no other processes running.

8| In WEKA, load the .arff file in the 'Preprocess' tab (**Fig. 3a**) and then switch to 'Classify' (**Fig. 3b**). Select classifiers in the 'Classifier - Choose' option: first select CostSensitiveClassifier under 'meta' classifiers. Then, select in the popup window Bagging under 'meta' classifiers. Click on the text bar listing Bagging and select RandomForest under 'meta' classifiers. Within the popup window for the CostSensitiveClassifier, define a 'costMatrix'. Cost-sensitive training is crucial as the training dataset is heavily biased toward one class (e.g., here 91% of *nonobserved* peptides) and a cost matrix counteracts this bias by weighted use of the training data. In our example, the cost matrix looks like:

0.00	0.91
0.09	0.00

The cost matrix can also be saved and uploaded in later uses. Specify 10 in the Cross-validation.

▲ CRITICAL STEP If no cost matrix is specified, model performance is very poor, in particular if there is a strong class bias in training data (see **Supplementary Note** online, Table S2). In fact, we recommend reversing or leaving-out the cost matrix as a control experiment: decreasing model performance (*F*-measure) compared to correct use of a cost matrix verifies setup of the calculations. In contrast, classifiers other than Bagging and RandomForests can also perform well, as discussed in the original APEX publication[25].

9| Start calculations by clicking on 'Start'. Depending on computer power and dataset size model building and cross-validation takes several minutes.

10| The output file contains information on the success of the training (**Fig. 3c**), for example via the *F*-measure which is the weighted harmonic mean of precision and recall [2 × precision × recall/(precision + recall)] of class prediction. The closer the *F*-measure is to 1, the larger are precision and recall and the better is the prediction. In many training sets, most peptides are *not* observed; prediction of peptide observation is harder than prediction of nonobservation. Therefore, we recommend paying special attention to the *F*-measure (as well as precision, recall) of *observed* peptides (class 1); the larger this *F*-measure, the better is the model. We recommend an *F*-measure of >0.5 (empirical evidence).

In the example (**Supplementary Note** online), observed peptides (class 1) are predicted with an *F*-measure of 0.61, i.e., with precision and recall of 0.59 and 0.63, respectively. Nonobserved peptides (class 0) are predicted with much higher precision (0.96) and recall (0.96), and the *F*-measure is 0.96.

11| Once the training is over and a quality model has been created, save the model as a .model file by right-clicking in the 'Results list' section and selecting 'Save model'.

■ PAUSE POINT The saved model can be used at any later time and in different experiments.

Training and testing of a model for prediction of peptide and protein MS detectability: testing

12| Postprocess MS/MS raw data as in Step 1 and **Figure 2a**. This time include all proteins of interest, e.g., with $<5\%$ false discovery rate.

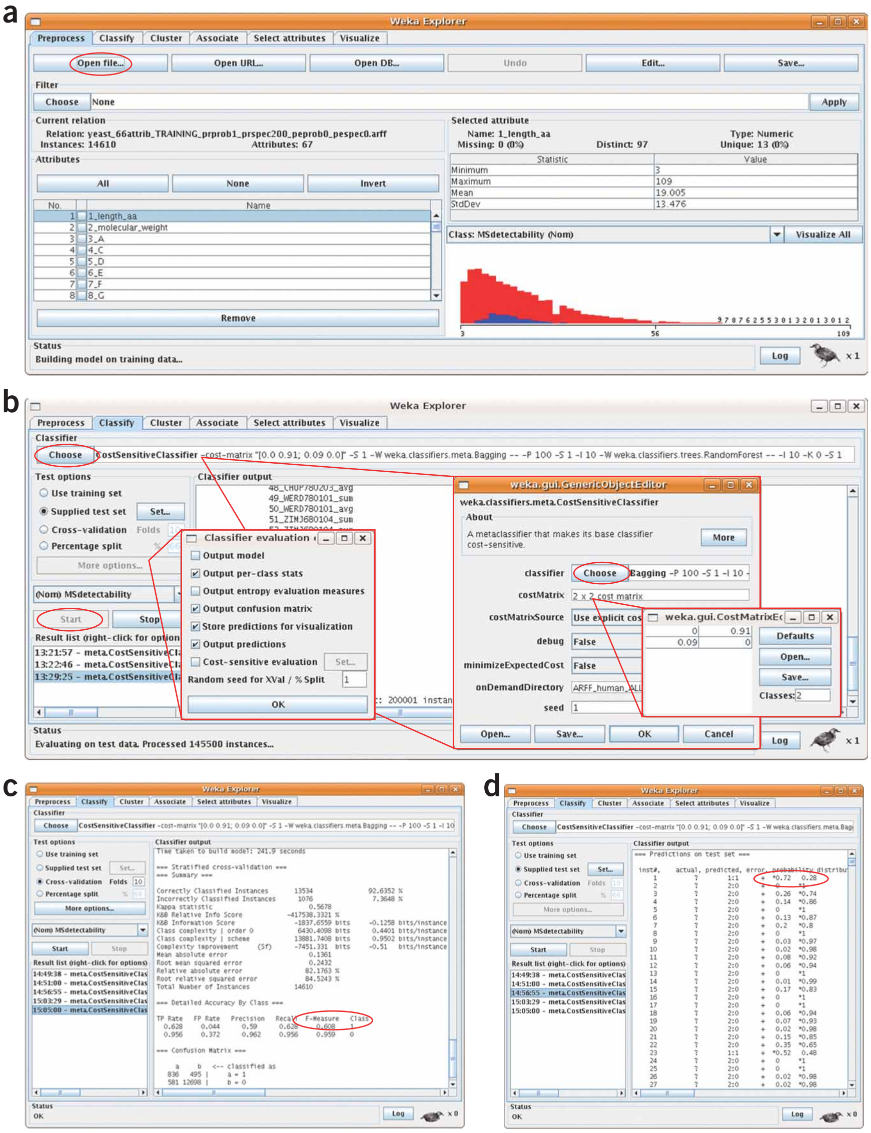
13| Digest the amino acid sequences for all proteins of interest (above) *in silico* into (tryptic) peptides, using the same parameters as in Step 3, i.e., allow for the same number of missed cleavages. This file easily becomes large; a yeast genome with ~6,000 genes *in silico* digests into ~921,000 peptides (≤2 missed cleavages).

14| Analyze all peptides for their sequence features using the same attributes as in Step 4.

15| Convert peptide feature vectors into WEKA .arff file format similar to Step 6. At the end of each feature vector, place a question mark '?' instead of the '1' or '0' describing peptide observation (**Fig. 2b**).

16| Predict probability of observation (peptide MS detectability) using WEKA. In the 'Preprocess' tab, load the .arff file created in Step 4. In the 'Classify' tab, load the model created in 'Training and testing of a model for prediction of peptide and protein MS detectability: training' by right-clicking within the 'Result list' section and choosing 'Load model'. If you do not yet have a model available, create it according to 'Training and testing of a model for prediction of peptide and protein MS detectability: training'. Select CostSensitiveClassifier, Bagging and Random Forests as classifiers and defined a cost matrix as described in Step 8. Do *not* select Cross-validation. Select the 'Supplied test set' option and upload the test .arff file. Under 'More Options', unselect to output the model and select to display the output predictions.

17| Start calculations by clicking on 'Start'. Depending on computer power and dataset size the calculations can take several minutes.



Current relation
Relation: yeast_66attrib_TRAINING_prprob1_prspec200_peprob0_pespec0.arff
Instances: 14610 Attributes: 67

Selected attribute
Name: 1_length_aa Type: Numeric
Missing: 0 (0%) Distinct: 97 Unique: 13 (0%)

Statistic	Value
Minimum	3
Maximum	109
Mean	19.005
StdDev	13.476

Attributes

No.	Name
1	1_length_aa
2	2_molecular_weight
3	3_A
4	4_C
5	5_D
6	6_E
7	7_F
8	8_G

Class: MSdetectability (Nom)

Status
Building model on training data...

Classifier: CostSensitiveClassifier -cost-matrix "[0.0 0.91; 0.09 0.0]" -S 1 -W weka.classifiers.meta.Bagging -- -P 100 -S 1 -I 10 -W weka.classifiers.trees.RandomForest -- -I 10 -K 0 -S 1

Test options
○ Use training set
● Supplied test set
○ Cross-validation Folds 10
○ Percentage split % 66

(Nom) MSdetectability

Result list (right-click for options)
13:21:57 - meta.CostSensitiveCla
13:22:46 - meta.CostSensitiveCla
13:29:25 - meta.CostSensitiveCla

Classifier output
```
48_CHOP780203_avg
49_WERD780101_sum
50_WERD780101_avg
51_ZIMJ680104_sum
...
: 200001 instan
```

Classifier evaluation
☐ Output model
☑ Output per-class stats
☐ Output entropy evaluation measures
☑ Output confusion matrix
☑ Store predictions for visualization
☑ Output predictions
☐ Cost-sensitive evaluation
Random seed for XVal / % Split: 1

weka.classifiers.meta.CostSensitiveClassifier
About
A metaclassifier that makes its base classifier cost-sensitive.

classifier: Bagging -P 100 -S 1 -I 10 -
costMatrix: 2 x 2 cost matrix
costMatrixSource: Use explicit cos
debug: False
minimizeExpectedCost: False
onDemandDirectory: ARFF_human_ALL
seed: 1

0	0.91
0.09	0

Classes: 2

Status
Evaluating on test data. Processed 145500 instances...

Classifier: CostSensitiveClassifier -cost-matrix "[0.0 0.91; 0.09 0.0]" -S 1 -W weka.classifiers.meta.Bagging -- -P 100 -S 1 -I 10

Test options
○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 66

Result list (right-click for options)
14:49:38 - meta.CostSensitiveClas
14:51:00 - meta.CostSensitiveClas
14:56:55 - meta.CostSensitiveClas
15:03:29 - meta.CostSensitiveClas
15:05:00 - meta.CostSensitiveClas

Classifier output
```
Time taken to build model: 241.9 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       13534              92.6352 %
Incorrectly Classified Instances      1076               7.3648 %
Kappa statistic                          0.5678
K&B Relative Info Score            -417538.3321 %
K&B Information Score               -1837.6559 bits     -0.1258 bits/instance
Class complexity | order 0           6430.4098 bits      0.4401 bits/instance
Class complexity | scheme           13881.7408 bits      0.9502 bits/instance
Complexity improvement     (Sf)     -7451.331  bits     -0.51   bits/instance
Mean absolute error                      0.1361
Root mean squared error                  0.2432
Relative absolute error                 82.1763 %
Root relative squared error             84.5243 %
Total Number of Instances            14610

=== Detailed Accuracy By Class ===

TP Rate   FP Rate   Precision   Recall  F-Measure   Class
  0.628     0.044     0.59        0.628     0.608     1
  0.956     0.372     0.962       0.956     0.959     0

=== Confusion Matrix ===

     a     b   <-- classified as
   836   495 |     a = 1
   581 12698 |     b = 0
```

Status
OK

Classifier: CostSensitiveClassifier -cost-matrix "[0.0 0.91; 0.09 0.0]" -S 1 -W weka.classifiers.meta.Bag

Test options
○ Use training set
● Supplied test set
○ Cross-validation Folds 10
○ Percentage split % 66

Result list (right-click for options)
14:49:38 - meta.CostSensitiveClas
14:51:00 - meta.CostSensitiveClas
14:56:55 - meta.CostSensitiveClas
15:03:29 - meta.CostSensitiveClas
15:05:00 - meta.CostSensitiveClas

Classifier output
```
=== Predictions on test set ===

 inst#,    actual, predicted, error, probability distribu
     1        ?        1:1      +   *0.72   0.28
     2        ?        2:0      +    0     *1
     3        ?        2:0      +    0.26  *0.74
     4        ?        2:0      +    0.14  *0.86
     5        ?        2:0      +    0     *1
     6        ?        2:0      +    0.13  *0.87
     7        ?        2:0      +    0.2   *0.8
     8        ?        2:0      +    0     *1
     9        ?        2:0      +    0.03  *0.97
    10        ?        2:0      +    0.02  *0.98
    11        ?        2:0      +    0.08  *0.92
    12        ?        2:0      +    0.06  *0.94
    13        ?        2:0      +    0     *1
    14        ?        2:0      +    0.01  *0.99
    15        ?        2:0      +    0.17  *0.83
    16        ?        2:0      +    0     *1
    17        ?        2:0      +    0     *1
    18        ?        2:0      +    0.06  *0.94
    19        ?        2:0      +    0.07  *0.93
    20        ?        2:0      +    0.02  *0.98
    21        ?        2:0      +    0.15  *0.85
    22        ?        2:0      +    0.35  *0.65
    23        ?        1:1      +   *0.52   0.48
    24        ?        2:0      +    0     *1
    25        ?        2:0      +    0     *1
    26        ?        2:0      +    0.02  *0.98
    27        ?        2:0      +    0.02  *0.98
```

Status
OK

Figure 3 | Use of WEKA. The screenshots illustrate how use of the WEKA Explorer can look like in practice. Red circles mark steps described in this protocol. (**a**) Uploading the .arff file. (**b**) Choosing the classifier and defining cost matrix and other parameters. (**c**) Training output. (**d**) Prediction output.

18| Cut and paste the output file into a text file or save it by right-clicking in the 'Result list' section and selecting 'Save result buffer'. The second but last column of the output file provides the probability of peptide observation (**Fig. 3d**), i.e., the class 1 probability, and this value is used for further calculations. Note that while peptide MS detectability is binary during training (*observed/non-observed*), it is continuous when calculating O_i (class 1 probability: value between 0 and 1).

19| Match the peptide identities to probabilities of peptide observation of the WEKA output file. Sum over the probabilities for all peptides mapping to a protein; this sum is the O_i-value of the protein, i.e., the *expected* number of observed peptides. Store these O_i-values in a data file.

Once calculated for an organism for a particular experimental setup, the O_i-values can be reused for any number of MS/MS analyses of the same proteins. See http://www.marcottelab.org/APEX_Protocol/ for O_i-values for the entire proteomes of *E. coli*, yeast and human for analysis on an LCQ and an LTQ-OrbiTrap using a given protocol, mass range, etc. (provided on the website).

■ **PAUSE POINT** O_i-values can be used at any later time and in different experiments involving the same proteins.

Estimation of absolute protein expression levels

20| Postprocess MS/MS raw data as described in Step 1. For each protein identified in the MS/MS experiment, we need the probability of correct identification p_i and the total number of observed MS/MS spectra n_i.

21| Calculate O_i-values for each protein as described in Steps 12–19, i.e., the expected number of unique peptides per protein corrected by the differential peptide MS detectability.

22| Estimate the total number of protein molecules per cell C. A total of 5×10^7 molecules/cell[28] for yeast and $2\text{–}3 \times 10^6$ molecules/cell for *E. coli*[36,37] have been suggested. This total number of molecules will be split among the proteins identified in the MS experiment. Because the number of proteins identified can vary between different experiments, an alternative way to estimate C is to multiply the number of proteins identified by an estimate of the average number of molecules per protein. For yeast, an average of 2,000 to 10,000 molecules per protein is expected[25,26,28], for *E. coli* ~1,000 to 4,000 (refs. 25,36,37). In our example experiment, 2,033 proteins were identified with <5% false discovery rate on the LTQ-OrbiTrap, thus we estimate $C = 2{,}033 \times 4{,}000 \approx 8.1 \times 10^6$. Alternatively, if not cellular lysates but a protein mixture is used, C can be estimated using the total concentration of proteins in the sample. Finally, C can also be set to a constant (e.g., 1) which results in APEX values of proteins *relative* to each other in the sample. The protein abundances from the last estimate cannot be compared between samples.

23| Calculate APEX protein absolute protein expression values using equation (1).

$$\mathrm{APEX}_i = \frac{n_i \times p_i}{O_i \times \sum_{k=1}^{\text{No. of observed proteins}} n_k \times p_k / O_k} \times C \qquad (1)$$

In equation (1), n_i is the total spectral count for protein i (total number of MS/MS spectra attributable to protein i), O_i is the expected unique peptide count for protein i (sum of peptide MS detectabilities for a given protein), and p_i is the protein identification probability. Values for n_i and p_i are extracted from postprocessed MS/MS data; O_i is computed as described earlier.

As an overall control for correct APEX calculations, we recommend that the user conducts a spike-in experiment as described in the original publication[25]. In such an experiment, a mixture of proteins of known abundances is spiked into cellular lysate and APEX is used to estimate protein concentrations in the mixture.

Estimation of relative protein expression (comparison of two samples)

24| Postprocess MS/MS raw data of both samples as described in Step 1. For each protein identified in the MS/MS experiment, we need the probability of correct identification p_i and the total number of observed MS/MS spectra n_i.

25| Calculate the total number of observed MS/MS spectra (total spectral counts) N for each sample. This sum includes only peptides of confident identification (above threshold). Convert the spectral counts n_i into fractions $f_i = n_i/N$.

26| Calculate for each protein the overall proportion $f_{i,0} = (n_{i,1} + n_{i,2})/(N_1 + N_2)$. The proportion $f_{i,0}$ is the null expectation in the event that protein i is present at the same level in both samples. The calculation can be done for proteins which are confidently identified in *both* samples, and for proteins which are only identified in *one* sample but assumed to be absent in the other sample.

27| Calculate APEX-based protein abundance estimates as described in Steps 20–23. The expression fold change between the two samples 1 and 2 can then be expressed as the ratio $\mathrm{APEX}_{i,1}/\mathrm{APEX}_{i,2}$. If a protein is absent in one sample, its spectral count is $n_i = 0$ and an APEX-based fold change cannot be calculated.

28| Calculate for each protein a Z-score of differential expression according to

$$Z = \frac{f_{i,1} - f_{i,2}}{\sqrt{f_{i,0}(1 - f_{i,0})/N_1 + f_{i,0}(1 - f_{i,0})/N_2}} \qquad (2)$$

where N_1 and N_2 are the total spectral counts in samples 1 and 2, $f_{i,0}$ is the overall proportion of a protein's spectral counts (Step 26), and $f_{i,1}$ and $f_{i,2}$ are the proportions of a protein's spectral counts in sample 1 and 2, respectively (Step 25).

Equation (2) is based on a similar approach applied in SAGE mRNA expression profiling[38–40]. Two-sided P values require $|Z| > 1.96$ for P value < 0.05; $|Z| > 2.58$ for P value < 0.01. Proteins of high abundance in both samples can be significantly differentially expressed even if the actual expression fold change is small. Thus we recommend examining both Z-scores and expression fold changes for each protein. The **Supplementary Note** online and the website at http://www.marcottelab.org/APEX_Protocol/ contain an example of differential protein expression analysis (yeast grown in minimal versus rich medium).

? TROUBLESHOOTING

● TIMING

Training and testing: a few minutes to several hours once scripts and data files are setup.
Estimation of absolute and relative expression values: minutes, once scripts and data files are set up.

? TROUBLESHOOTING

WEKA crashes during training or testing

WEKA explorer uses a lot of memory, especially when handling large files. If WEKA crashes during model building (training), consider allocating more memory or reducing dataset size by filtering the training data more stringently (see ▲ CRITICAL STEP, Step 2).

When applying the model to predict peptide MS detectability, we found that for a test file with 100,000 lines, at least 1,500 MB memory is required (allocated in Step 7). If the test file contains more than 100,000 lines, we recommend splitting the file into smaller .arff files, assigning more memory when starting WEKA (Step 7) and/or using the WEKA command line interface. The peptide file for the whole yeast genome needs to be split into ~10 separate .arff files with each 100,000 lines or fewer. Be sure to have unselected 'Output model' under 'More options' to save the memory required to output the model.

An error message appears when uploading the .arff training or testing file

Thoroughly check the .arff file format. Check that the number of attributes listed in the header is the same as the number of attributes (features) in the data rows. Ensure that all rows with data entries have the same number of attributes listed. If nothing helps, try uploading our example .arff files and work from there.

Training results in a poor model, e.g., the *F*-measure for *observed* peptides is $\ll 0.5$

Check that the correct cost matrix is used, as described in Step 8. Check quality of the training data (CRITICAL STEP, Step 2). Consider reducing your training set to fewer proteins, possibly hand-select them for their quality of peptide identification. Check that peptides classified as observed have high-peptide identification scores (or probabilities). Check that proteins in the training set are not degenerate, i.e., that several proteins of different names do not map to the same group of peptides. Check that peptides in the training set are not degenerate, i.e., that their observation is not mapped to several proteins of different names. (When selecting our training data, we exclude all degenerate proteins and peptides.) Ensure that you use WEKA correctly by training on one of the files provided at http://www.marcottelab.org/APEX_Protocol/ and comparing your training outputs with our result files.

Check types of peptide attributes (CRITICAL STEP, Step 4). Modify the kinds and number of attributes used to describe peptide sequences. Not all 66 attributes used in our example set are equally important for training. Performing different tests in the 'Attribute selection' section in WEKA (Ranker-PrincipalComponents, Ranker-InfoGain and BestFirst-CfsSubset), we identified attributes describing peptide length, the iso-electric point, hydrophobicity, solvent access, solubility, volume, secondary structure as most important, while amongst amino acid frequencies the number of C, R and K were top-ranked (see http://www.marcottelab.org/APEX_Protocol/). Consider adding attributes listed by Mallick *et al.*[23] as important for your experimental setup (if not yet included).

ANTICIPATED RESULTS

In our example analysis, we train prediction of peptide MS detectability on a set of 89 yeast proteins well-observed in an LTQ-OrbiTrap MS/MS experiment and then estimate O_i-values for all proteins in the yeast genome (**Supplementary Note** online and http://www.marcottelab.org/APEX_Protocol/). As an example, the TFC3 protein (YAL001C) has ~500 theoretical peptides from a tryptic digest with ≤ 2 missed cleavages. Only four different peptides are observed in the experiment with five spectral counts. Given sequence properties of all peptides, TFC3's O_i is 60.24, i.e., about 60 peptides are expected for this protein to be observed in an LC-MS/MS analysis. With an average of 4,000 molecules/protein and 2,033 proteins detected in total, the APEX value for TFC3 is estimated to 116 molecules/cell.

When establishing the APEX protocol we encourage the reader to use the Perl scripts and sample data files provided on our website as a control for correct setup. Further, probabilities of peptide MS detectability may also be compared to predictions by Mallick *et al.*[23] and by the Peptide Detectability Predictor at http://darwin.informatics.indiana.edu/applications/PeptideDetectabilityPredictor/. Other tests of the quality of APEX estimates are described in the original publication[25].

We provide this protocol not only for easy calculation of absolute and relative protein expression values but also to encourage the reader to experiment and optimize the method to suit his or her needs. In future work, several refinements are possible. For example, when training for peptide MS detectability, actual peptide identification probabilities could be taken into account, converting the binary classification (*observed*, *nonobserved*) into a continuous value. Peptide charge states and prior modifications (e.g., on Cysteine residues) may also be considered. Further, the user may choose to allow only ≤ 1 missed cleavages instead of 2.

APEX will also be applied in a free software tool which is being developed by John Braisted and colleagues at the J. Craig Venter Institute (JCVI), Rockville, MD (J.C. Braisted, personal communication).

Note: Supplementary information is available via the HTML version of this article.

ACKNOWLEDGMENTS C.V. acknowledges support by the International Human Frontier Science Program. We thank John Braisted and Srilatha Kuntumalla from JCVI for many useful discussions regarding the APEX calculations. This work was supported by grants from the Welch (F-1515) and Packard Foundations, the National Science Foundation and National Institutes of Health.

Published online at http://www.natureprotocols.com/
Reprints and permissions information is available online at http://npg.nature.com/reprintsandpermissions/

1. Nesvizhskii, A.I., Keller, A., Kolker, E. & Aebersold, R. A statistical model for identifying proteins by tandem mass spectrometry. *Anal. Chem.* **75**, 4646–4658 (2003).
2. Silva, J.C., Gorenstein, M.V., Li, G.Z., Vissers, J.P. & Geromanos, S.J. Absolute quantification of proteins by LCMSE: a virtue of parallel MS acquisition. *Mol. Cell. Proteomics* **5**, 144–156 (2006).
3. Oda, Y., Huang, K., Cross, F.R., Cowburn, D. & Chait, B.T. Accurate quantitation of protein expression and site-specific phosphorylation. *Proc. Natl. Acad. Sci. U.S.A.* **96**, 6591–6596 (1999).
4. Ong, S.E. & Mann, M. Mass spectrometry-based proteomics turns quantitative. *Nat. Chem. Biol.* **1**, 252–262 (2005).
5. Ong, S.E. *et al.* Stable isotope labeling by amino acids in cell culture, SILAC, as a simple and accurate approach to expression proteomics. *Mol. Cell. Proteomics* **1**, 376–386 (2002).
6. Gygi, S.P. *et al.* Quantitative analysis of complex protein mixtures using isotope-coded affinity tags. *Nat. Biotechnol.* **17**, 994–999 (1999).
7. Gerber, S.A., Rush, J., Stemman, O., Kirschner, M.W. & Gygi, S.P. Absolute quantification of proteins and phosphoproteins from cell lysates by tandem MS. *Proc. Natl. Acad. Sci. U.S.A.* **100**, 6940–6945 (2003).
8. Ishihama, Y. *et al.* Quantitative mouse brain proteomics using culture-derived isotope tags as internal standards. *Nat. Biotechnol.* **23**, 617–621 (2005).
9. Liu, H., Sadygov, R.G. & Yates, J.R. 3rd. A model for random sampling and estimation of relative protein abundance in shotgun proteomics. *Anal. Chem.* **76**, 4193–4201 (2004).
10. Gao, J., Opiteck, G.J., Friedrichs, M.S., Dongre, A.R. & Hefta, S.A. Changes in the protein expression of yeast as a function of carbon source. *J. Proteome Res.* **2**, 643–649 (2003).
11. Florens, L. *et al.* A proteomic view of the Plasmodium falciparum life cycle. *Nature* **419**, 520–526 (2002).
12. Gao, J., Friedrichs, M.S., Dongre, A.R. & Opiteck, G.J. Guidelines for the routine application of the Peptide hits technique. *J. Am. Soc. Mass. Spectrom.* **16**, 1231–1238 (2005).
13. States, D.J. *et al.* Challenges in deriving high-confidence protein identifications from data gathered by a HUPO plasma proteome collaborative study. *Nat. Biotechnol.* **24**, 333–338 (2006).
14. Blondeau, F. *et al.* Tandem MS analysis of brain clathrin-coated vesicles reveals their critical involvement in synaptic vesicle recycling. *Proc. Natl. Acad. Sci. U.S.A.* **101**, 3833–3838 (2004).
15. Kislinger, T. *et al.* Global survey of organ and organelle protein expression in mouse: combined proteomic and transcriptomic profiling. *Cell* **125**, 173–186 (2006).
16. Kislinger, T. *et al.* Proteome dynamics during C2C12 myoblast differentiation. *Mol. Cell Proteomics* **4**, 887–901 (2005).
17. Steen, H. & Pandey, A. Proteomics goes quantitative: measuring protein abundance. *Trends Biotechnol.* **20**, 361–364 (2002).
18. Elias, J.E., Gibbons, F.D., King, O.D., Roth, F.P. & Gygi, S.P. Intensity-based protein identification by machine learning from a library of tandem mass spectra. *Nat. Biotechnol.* **22**, 214–219 (2004).
19. Gay, S., Binz, P.A., Hochstrasser, D.F. & Appel, R.D. Peptide mass fingerprinting peak intensity prediction: extracting knowledge from spectra. *Proteomics* **2**, 1374–1391 (2002).
20. Craig, R., Cortens, J.P. & Beavis, R.C. The use of proteotypic peptide libraries for protein identification. *Rapid Commun. Mass. Spectrom.* **19**, 1844–1850 (2005).
21. Kuster, B., Schirle, M., Mallick, P. & Aebersold, R. Scoring proteomes with proteotypic peptide probes. *Nat. Rev. Mol. Cell Biol.* **6**, 577–583 (2005).
22. Le Bihan, T., Robinson, M.D., Stewart, I.I. & Figeys, D. Definition and characterization of a "trypsinosome" from specific peptide characteristics by nano-HPLC-MS/MS and *in silico* analysis of complex protein mixtures. *J. Proteome Res.* **3**, 1138–1148 (2004).
23. Mallick, P. *et al.* Computational prediction of proteotypic peptides for quantitative proteomics. *Nat. Biotechnol.* **25**, 125–131 (2007).
24. Tang, H. *et al.* A computational approach toward label-free protein quantification using predicted peptide detectability. *Bioinformatics* **22**, e481–e488 (2006).
25. Lu, P., Vogel, C., Wang, R., Yao, X. & Marcotte, E.M. Absolute protein expression profiling estimates the relative contributions of transcriptional and translational regulation. *Nat. Biotechnol.* **25**, 117–124 (2007).
26. Ghaemmaghami, S. *et al.* Global analysis of protein expression in yeast. *Nature* **425**, 737–741 (2003).
27. Newman, J.R. *et al.* Single-cell proteomic analysis of *S. cerevisiae* reveals the architecture of biological noise. *Nature* **441**, 840–846 (2006).
28. Futcher, B., Latter, G.I., Monardo, P., McLaughlin, C.S. & Garrels, J.I. A sampling of the yeast proteome. *Mol. Cell. Biol.* **19**, 7357–7368 (1999).
29. Lopez-Campistrous, A. *et al.* Localization, annotation, and comparison of the *Escherichia coli* K-12 proteome under two states of growth. *Mol. Cell Proteomics* **4**, 1205–1209 (2005).
30. Lu, P. *et al.* Global metabolic changes following loss of a feedback loop reveal dynamic steady states of the yeast metabolome. *Metab. Eng.* **9**, 8–20 (2007).
31. Wang, R. & Marcotte, E.M. The proteomic response of *Mycobacterium smegmatis* to anti-tuberculosis drugs suggests targeted pathways. *J. Proteome Res.* **7**, 855–865 (2008).
32. Baerenfaller, K. *et al.* Genome-scale proteomics reveals *Arabidopsis thaliana* gene models and proteome dynamics. *Science* **320**, 938–941 (2008).
33. Schmidt, M.W., Houseman, A., Ivanov, A.R. & Wolf, D.A. Comparative proteomic and transcriptomic profiling of the fission yeast *Schizosaccharomyces pombe*. *Mol. Syst. Biol.* **3**, 79 (2007).
34. Keller, A., Nesvizhskii, A.I., Kolker, E. & Aebersold, R. Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search. *Anal. Chem.* **74**, 5383–5392 (2002).
35. Cagney, G., Amiri, S., Premawaradena, T., Lindo, M. & Emili, A. *In silico* proteome analysis to facilitate proteomics experiments using mass spectrometry. *Proteome Sci.* **1**, 5 (2003).
36. Neidhardt, F.C. & Umbarger, H.E. In *Escherichia coli and Salmonella typhimurium: Cellular and Molecular Biology* 2nd edn. Vol. 1 (eds. Neidhardt, F.C. *et al.*) 13–16 (ASM Press, Washington, D.C., 1996).
37. Sundararaj, S. *et al.* The CyberCell Database (CCDB): a comprehensive, self-updating, relational database to coordinate and facilitate *in silico* modeling of *Escherichia coli*. *Nucleic Acids Res.* **32**, D293–D295 (2004).
38. Kal, A.J. *et al.* Dynamics of gene expression revealed by comparison of serial analysis of gene expression transcript profiles from yeast grown on two different carbon sources. *Mol. Biol. Cell* **10**, 1859–1872 (1999).
39. Stollberg, J., Urschitz, U., Urban, Z. & Boyd, C.D. A quantitative evaluation of SAGE. *Genome Res.* **10**, 1241–1248 (2000).
40. Velculescu, V.E., Zhang, L., Vogelstein, B. & Kinzler, K.W. Serial analysis of gene expression. *Science* **270**, 484–487 (1995).