demo_perm

```
def demo_perm_glynn(M):
    col_sums = [sum(c) for c in zip(*M)]
    n = M.dimensions()[0]
    total = 0
    old_gray = 0
    d_i = +1
    powers_of_2_dict = {2**i:i for i in range(n)}
    num_loops = 2**(n-1)

    for num in xrange(1, num_loops + 1):
        prod = 1
        print "iteration:",num, "colsums: ",col_sums
        for i in col_sums:
            prod *= i
        print "prod of col_sums:",prod
        total += d_i * prod
        print "total:", total

        new_gray = num ^^ (num >> 1)
        diff = old_gray ^^ new_gray
        diff_index = powers_of_2_dict[diff]
        new_vector = M.row(diff_index)
        dir = 2 * cmp(old_gray,new_gray)

        for i in range(n):
            col_sums[i] += new_vector[i] * dir

        d_i = -d_i
        old_gray = new_gray

    return total / num_loops
```

```
def demo_perm_spies(M):


    row_sums = [sum(r) for r in M]
    n = M.dimensions()[0]
    total = 0
    old_gray = 0
    x_i = +1
    powers_of_2_dict = {2**i:i for i in range(n)}
    num_loops = 2**(n-1)

    for num in xrange(1, num_loops + 1):
        prod = 1
        print "iteration:",num, "rowsums: ",row_sums
        for x in row_sums:
            prod *= x
        print "prod of row_sums:",prod
        total += x_i * prod
        print "total:", total
        new_gray = num ^^ (num >> 1)
        diff = old_gray ^^ new_gray
        diff_index = powers_of_2_dict[diff]
        new_vector = M.column(diff_index)
        dir = 2 * cmp(old_gray,new_gray)

        for i in range(n):
            row_sums[i] += new_vector[i] * dir

        x_i = -x_i
        old_gray = new_gray

    return total / num_loops
```

```
R.<a,b,c,d,e,f,g,h,i> = PolynomialRing(ZZ)
A = matrix(R,3,3,[a,b,c,d,e,f,g,h,i])
```

```
demo_perm_glynn(A)
```

```
iteration: 1 colsums:  [a + d + g, b + e + h, c + f + i]
prod of col_sums: a*b*c + b*c*d + a*c*e + c*d*e + a*b*f + b*d*f +
a*e*f + d*e*f + b*c*g + c*e*g + b*f*g + e*f*g + a*c*h + c*d*h +
a*f*h + d*f*h + c*g*h + f*g*h + a*b*i + b*d*i + a*e*i + d*e*i +
b*g*i + e*g*i + a*h*i + d*h*i + g*h*i
total: a*b*c + b*c*d + a*c*e + c*d*e + a*b*f + b*d*f + a*e*f + d*e*f
+ b*c*g + c*e*g + b*f*g + e*f*g + a*c*h + c*d*h + a*f*h + d*f*h +
c*g*h + f*g*h + a*b*i + b*d*i + a*e*i + d*e*i + b*g*i + e*g*i +
a*h*i + d*h*i + g*h*i
iteration: 2 colsums:  [-a + d + g, -b + e + h, -c + f + i]
prod of col_sums: -a*b*c + b*c*d + a*c*e - c*d*e + a*b*f - b*d*f -
a*e*f + d*e*f + b*c*g - c*e*g - b*f*g + e*f*g + a*c*h - c*d*h -
a*f*h + d*f*h - c*g*h + f*g*h + a*b*i - b*d*i - a*e*i + d*e*i -
b*g*i + e*g*i - a*h*i + d*h*i + g*h*i
total: 2*a*b*c + 2*c*d*e + 2*b*d*f + 2*a*e*f + 2*c*e*g + 2*b*f*g +
2*c*d*h + 2*a*f*h + 2*c*g*h + 2*b*d*i + 2*a*e*i + 2*b*g*i + 2*a*h*i
iteration: 3 colsums:  [-a - d + g, -b - e + h, -c - f + i]
prod of col_sums: -a*b*c - b*c*d - a*c*e - c*d*e - a*b*f - b*d*f -
a*e*f - d*e*f + b*c*g + c*e*g + b*f*g + e*f*g + a*c*h + c*d*h +
a*f*h + d*f*h - c*g*h - f*g*h + a*b*i + b*d*i + a*e*i + d*e*i -
b*g*i - e*g*i - a*h*i - d*h*i + g*h*i
total: a*b*c - b*c*d - a*c*e + c*d*e - a*b*f + b*d*f + a*e*f - d*e*f
+ b*c*g + 3*c*e*g + 3*b*f*g + e*f*g + a*c*h + 3*c*d*h + 3*a*f*h +
```

```
d*f*h + c*g*h - f*g*h + a*b*i + 3*b*d*i + 3*a*e*i + d*e*i + b*g*i -
e*g*i + a*h*i - d*h*i + g*h*i
iteration: 4 colsums:  [a - d + g, b - e + h, c - f + i]
prod of col_sums: a*b*c - b*c*d - a*c*e + c*d*e - a*b*f + b*d*f +
a*e*f - d*e*f + b*c*g - c*e*g - b*f*g + e*f*g + a*c*h - c*d*h -
a*f*h + d*f*h + c*g*h - f*g*h + a*b*i - b*d*i - a*e*i + d*e*i +
b*g*i - e*g*i + a*h*i - d*h*i + g*h*i
total: 4*c*e*g + 4*b*f*g + 4*c*d*h + 4*a*f*h + 4*b*d*i + 4*a*e*i
c*e*g + b*f*g + c*d*h + a*f*h + b*d*i + a*e*i
```

```
demo_perm_spies(A)
```

```
iteration: 1 rowsums:  [a + b + c, d + e + f, g + h + i]
prod of row_sums: a*d*g + b*d*g + c*d*g + a*e*g + b*e*g + c*e*g +
a*f*g + b*f*g + c*f*g + a*d*h + b*d*h + c*d*h + a*e*h + b*e*h +
c*e*h + a*f*h + b*f*h + c*f*h + a*d*i + b*d*i + c*d*i + a*e*i +
b*e*i + c*e*i + a*f*i + b*f*i + c*f*i
total: a*d*g + b*d*g + c*d*g + a*e*g + b*e*g + c*e*g + a*f*g + b*f*g
+ c*f*g + a*d*h + b*d*h + c*d*h + a*e*h + b*e*h + c*e*h + a*f*h +
b*f*h + c*f*h + a*d*i + b*d*i + c*d*i + a*e*i + b*e*i + c*e*i +
a*f*i + b*f*i + c*f*i
iteration: 2 rowsums:  [-a + b + c, -d + e + f, -g + h + i]
prod of row_sums: -a*d*g + b*d*g + c*d*g + a*e*g - b*e*g - c*e*g +
a*f*g - b*f*g - c*f*g + a*d*h - b*d*h - c*d*h - a*e*h + b*e*h +
c*e*h - a*f*h + b*f*h + c*f*h + a*d*i - b*d*i - c*d*i - a*e*i +
b*e*i + c*e*i - a*f*i + b*f*i + c*f*i
total: 2*a*d*g + 2*b*e*g + 2*c*e*g + 2*b*f*g + 2*c*f*g + 2*b*d*h +
2*c*d*h + 2*a*e*h + 2*a*f*h + 2*b*d*i + 2*c*d*i + 2*a*e*i + 2*a*f*i
iteration: 3 rowsums:  [-a - b + c, -d - e + f, -g - h + i]
prod of row_sums: -a*d*g - b*d*g + c*d*g - a*e*g - b*e*g + c*e*g +
a*f*g + b*f*g - c*f*g - a*d*h - b*d*h + c*d*h - a*e*h - b*e*h +
c*e*h + a*f*h + b*f*h - c*f*h + a*d*i + b*d*i - c*d*i + a*e*i +
b*e*i - c*e*i - a*f*i - b*f*i + c*f*i
total: a*d*g - b*d*g + c*d*g - a*e*g + b*e*g + 3*c*e*g + a*f*g +
3*b*f*g + c*f*g - a*d*h + b*d*h + 3*c*d*h + a*e*h - b*e*h + c*e*h +
3*a*f*h + b*f*h - c*f*h + a*d*i + 3*b*d*i + c*d*i + 3*a*e*i + b*e*i
- c*e*i + a*f*i - b*f*i + c*f*i
iteration: 4 rowsums:  [a - b + c, d - e + f, g - h + i]
prod of row_sums: a*d*g - b*d*g + c*d*g - a*e*g + b*e*g - c*e*g +
a*f*g - b*f*g + c*f*g - a*d*h + b*d*h - c*d*h + a*e*h - b*e*h +
c*e*h - a*f*h + b*f*h - c*f*h + a*d*i - b*d*i + c*d*i - a*e*i +
b*e*i - c*e*i + a*f*i - b*f*i + c*f*i
total: 4*c*e*g + 4*b*f*g + 4*c*d*h + 4*a*f*h + 4*b*d*i + 4*a*e*i
c*e*g + b*f*g + c*d*h + a*f*h + b*d*i + a*e*i
```

```
```