Problem 5

In [22]:

```julia
# Julia only because numpy.polyfit produced unstable results
using Plots, LinearAlgebra, Statistics
```

In [23]:

```julia
w0, w1 = 1, 1

function interpolate(n, D)                               # I suppose it's
    α = sort!( 2 .* rand(n) .- 1 )                       # actually regression
    ỹ = w0 .+ w1 .* α                                    # undisturbed y
    y = ỹ + randn(n)                                     # disturbed y

    X = hcat( [α.^i for i in 0:D]... )
    Q, R = qr(X)                                         # qr factorization
    n > D+1 && ( R = vcat( R, zeros( n-(D+1), D+1 ) ) )  # for backward stability

    ŵ = R \ Q'y
    err = norm( X * ŵ - ỹ ) / n

    return err
end

interpolate(n, D, trials) = mean( interpolate(n, D) for _ in 1:trials )

x, y = 10:5:100, 1:5:90
xy = [(n, D) for D in y, n in x];
```

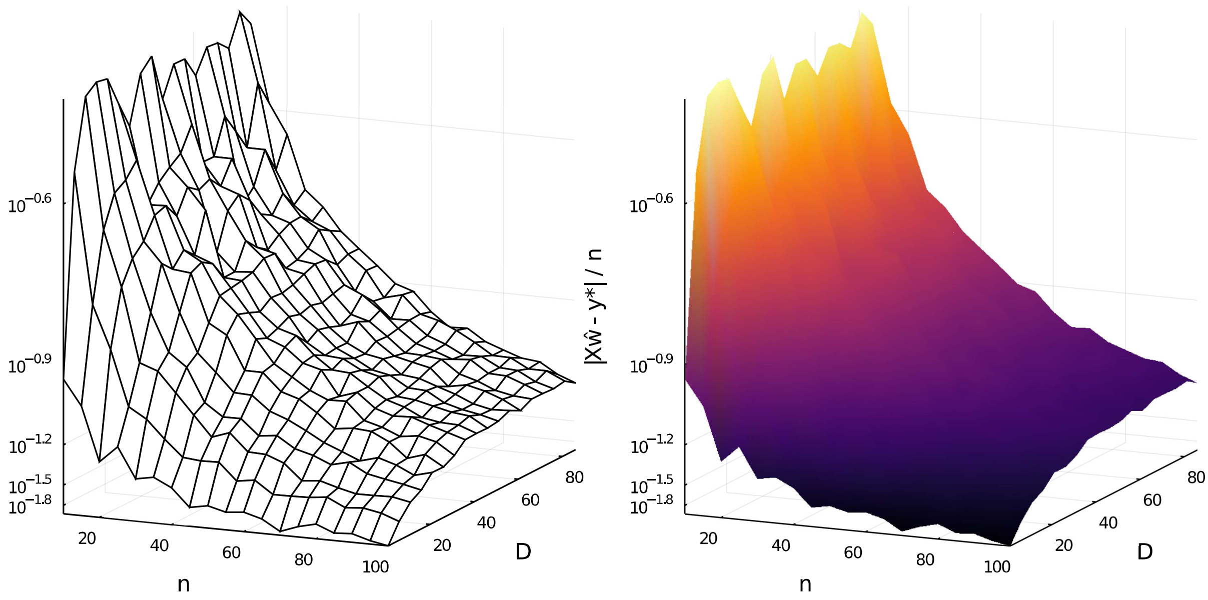
In [24]:

```julia
z = map(xy) do nD
    interpolate(nD..., 10)
end;
```

In [25]:

```julia
fig1 = wireframe(x, y, z,
                xaxis="n", yaxis="D", zaxis=:log10)
fig2 = surface(x, y, z,
               colorbar=false,
               xaxis="n", yaxis="D", zaxis=("|Xŵ - y*| / n", :log10))
fig = plot(fig1, fig2, layout=2, size=(900, 450))
```

Out[25]:



Problem 7

In [26]:

```julia
function interpolate(n, D)                           # redefine interpolate
    α = sort!( 7 .* rand(n) .- 4 )                   # function for exp(α)
    ỹ = exp.(α)                                      # undisturbed y
    y = ỹ + randn(n)                                 # disturbed y

    X = hcat( [α.^i for i in 0:D]... )
    Q, R = qr(X)                                     # qr factorization
    n > D+1 && ( R = vcat( R, zeros( n-(D+1), D+1 ) ) )

    ŵ = R \ Q'y
    err = norm( X * ŵ - ỹ ) / n

    return err
end;
```

In [27]:

```julia
x, y = 10:5:200, 1:50:201
xy = [(n, D) for n in x, D in y]
z = map(xy) do nD
    interpolate(nD..., 10)
end
fig1 = plot(x, collect(eachcol(z)),
            xaxis="n", yaxis=:log10,
            lab=permutedims(["constant D = $j" for j in y]));
```

In [37]:

```julia
x, y = [50, 120, 200], 10:5:200
xy = [(n, D) for D in y, n in x]
z = map(xy) do nD
    interpolate(nD..., 10)
end
fig2 = plot(y, collect(eachcol(z)),
            lab=permutedims(["constant n = $j" for j in x]),
            xaxis="D", yaxis=("|Xŵ - y*| / n", :log10),
            leg=:bottomright);
```

In [38]:

```julia
fig = plot(fig1, fig2, layout=2, size=(900, 450))
```

Out[38]: