```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Neural_Network
{
    class PerzeptronHiddenCell : Neuron
    {
        public override void learn(TrainingInstance ti) {
            setDelta(ti);

            foreach (Synapse s in incomingSynapses){
                s.weight += -learningRate * s.voltage * delta;
            }
        }

        public override void setDelta(TrainingInstance t) {
            double sumout = 0.0;
            foreach (Synapse s in outgoingSynapses) {
                s.to.calc();
                s.to.setDelta(t);
                sumout += s.weight * s.to.delta;
            }
            delta = activateDifferentiated(excitation()) * sumout;
        }

        protected override double activate(double sum)
        {
            //fermi function
            return 1 / (1 + Math.Pow(Math.E, -sum));
        }

        protected override double activateDifferentiated(double sum)
        {
            //differentiated fermi function
            double a = activate(sum);
            return a*(1.0-a);
        }
    }
}
```