# O Controlador Neuronal

O controlador neuronal tem como principal objetivo mimetizar o comportamento do controlador PID a uma dada entrada, ou seja, simulamos a planta industrial com um controlador PID normal, com

Como parâmetros do controlador PID.

Foi implementado no Simulink uma planta com controle PID com entrada randômica segundo a imagem abaixo:

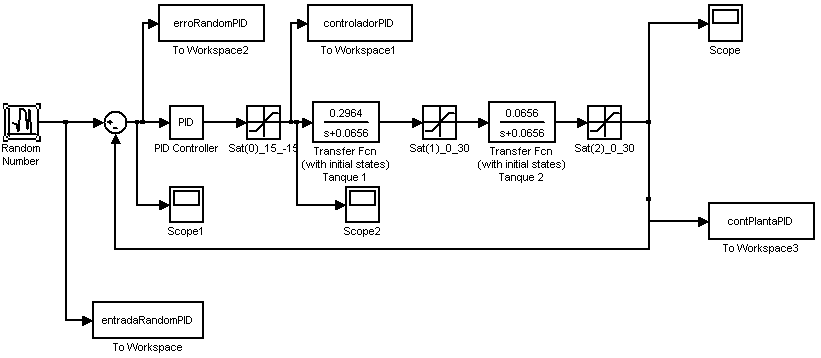


Figura - Planta simulada no Simulink

Com essa planta conseguimos simular o comportamento de uma planta industrial com controle feito por um controlador PID, dessa forma conseguimos adquirir informações de setpoint, erro e a saída do controlador PID. Com essas informações, fazemos com que a setpoint e o erro sejam a entrada de nossa rede neuronal e como alvo das validações da rede a saída do controlador neuronal.

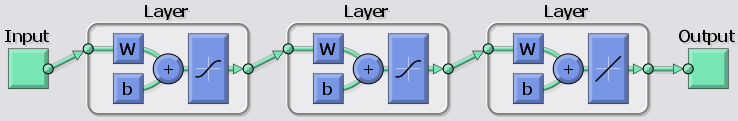


Figura - Arquitetura implementada

A arquitetura no projeto do controlador neuronal foi constituída de 3 camadas (imagem acima), sendo uma camada de entrada, uma camada oculta e uma camada de saída,10:10:1, com a camada de entrada e a camada oculta com função de ativação tangente-sigmoide e na camada de saída com uma função de ativação puramente linear. Utilizou-se uma rede feed-foward backpropagation, usando como algoritmo de treinamento o Levenberg-Marquardt ( função trainlm no Matlab), e com a seguinte seqüência de treinamento:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1º Treinamento |  | | Entrada: | Aleatória com média de 15V e variação de 15V | | Épocas: | **300** | | Performance: | **3.6925** | | 1treino_10-10-1_entradade15v_var_15.PNG |
| |  |  | | --- | --- | | 2º Treinamento |  | | Entrada: | Aleatória com média de 15V e variação de 15V | | Épocas: | **100** | | Performance: | **0.25047** | | primeiroTreino_arq_10-10-1.PNG |
| |  |  | | --- | --- | | 3º Treinamento |  | | Entrada: | Aleatória com média de 25V e variação de 5V | | Épocas: | **100** | | Performance: | **0.30094** | | 2treino_rede_10101.PNG |

Depois de todos os treinamentos efetuados, importamos a rede para o workspace e geramos o bloco referente ao Simulink com o comando gensim(Nome\_da\_redeNeural) e trocamos o controlador PID pelo bloco da rede neural gerado, segundo imagem abaixo:

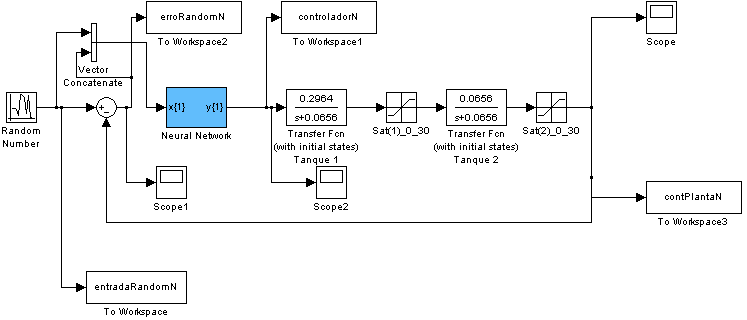


Figura - Planta com controlador neural

Na primeira comparação, tivemos o seguinte resultado, plotando a saída do controlador PID e do controlador neural:

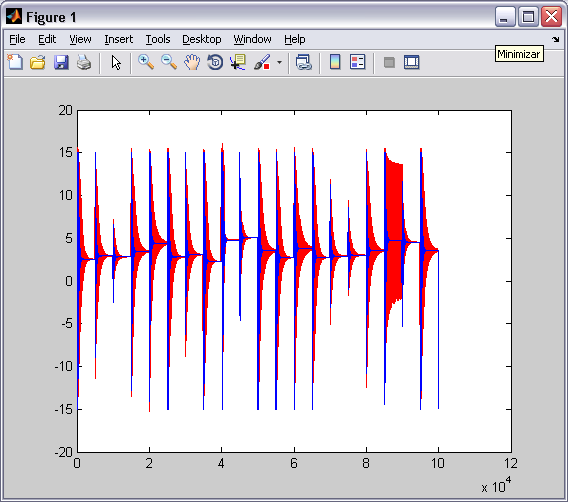


Figura - Comparação entre saida de controlador PID, em azul, e a saída do controlador neural em vermelho

Com o progresso dos treinos ficou explícito a melhora do sinal de saída do controlador neural, como na imagem a seguir:

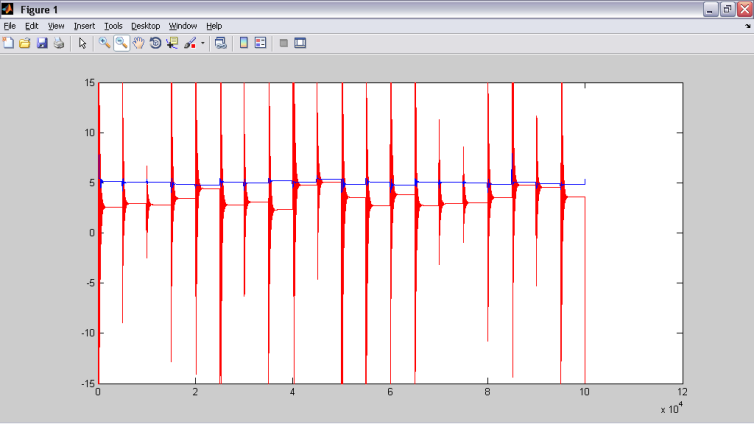


Figura - Comparação entre sinal do controlador PID e neural depois de sequencia de treinos

Durante os treinos, alguns dados alvos foram passados errados, resultando em uma completa “bagunça”

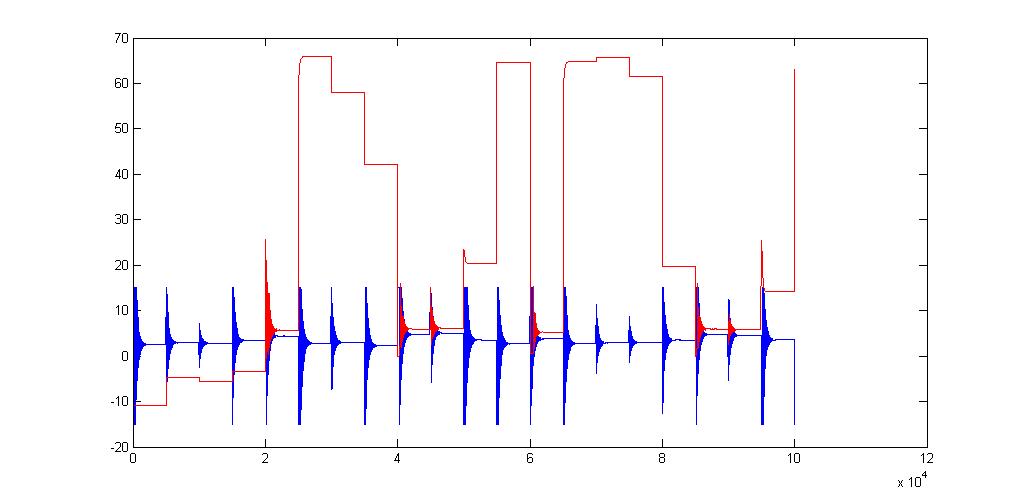


Figura - Resultado de entradas e saídas que não fazem parte da mesma parte de treino

A mudança de parâmetros de , e faria com que o overshoot, tempo de estabilização e etc, fossem mais interessantes para um melhor treinamento, visto que os parâmetros passados pelo professor para a simulação o controle do sinal ficou muito ruim, gerando um período transitório com overshoot muito elevado. Com a mudança de parâmetros de controle, conseguiríamos uma melhor performance com essa bateria de testes e dados.