A possible spatial entanglement measure

Detlef Schmicker

November 5, 2023

This is a work in progress: This is a part, which does something similar to what is the aim:

Taking a two particle wave function in one dimension $\psi(x, y)$ we suggest an entanglement measure. The wave function is not entangled, if it can be written in the form

$$\psi_{unentangled}(\overrightarrow{x}, y) = \phi_1(\overrightarrow{x})\phi_2(y) \ ,$$

and the density of state

$$\rho(\overrightarrow{x}, y) = \psi(\overrightarrow{x}, y) \, .$$

We can now compare two densities of states $\rho(\overrightarrow{x}, y_1)$ and $\rho(\overrightarrow{x}, y_2)$with the logarithm ratio:

$$r(\overrightarrow{x}, y_1, y_2) = \log\left(\frac{\rho(\overrightarrow{x}, y_1)}{\rho(\overrightarrow{x}, y_2)}\right) \ ,$$

and the weight with the density of states:

$$w(\overrightarrow{x}, y_1, y_2) = \rho(\overrightarrow{x}, y_1) + \rho(\overrightarrow{x}, y_2)$$

which would only depend on y_1 and y_2 in the case of no entanglement. This weighted logarithm will have a mean value and a standard derivation.

From this we can define a mean value

$$m = \frac{\int_{y_1 y_2}\left(\frac{\int_{\overrightarrow{x}} r(\overrightarrow{x}, y_1, y_2)\cdot w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}}{\int_{\overrightarrow{x}} w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}}\right) dy_1 dy_2}{\int_{\overrightarrow{x}\, y_1 y_2} w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}\, dy_1 dy_2} \ ,$$

and the standard deviation

$$s = \frac{\int_{y_1 y_2}\left(\left(\frac{\int_{\overrightarrow{x}} r^2(\overrightarrow{x}, y_1, y_2)\cdot w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}}{\int_{\overrightarrow{x}} w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}} - \left(\frac{\int_{\overrightarrow{x}} r(\overrightarrow{x}, y_1, y_2)\cdot w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}}{\int_{\overrightarrow{x}} w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}}\right)^2\right)\cdot \int_{\overrightarrow{x}} w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}\right) dy_1 dy_2}{\int_{\overrightarrow{x}\, y_1 y_2} w(\overrightarrow{x}, y_1, y_2) d\overrightarrow{x}\, dy_1 dy_2} \ ,$$

```
def get_ratio(wf, x, kk):
    """
    wf: wavefunction
    xx: The variable, nParticles-1 nParticles-1 and one variable at pos kk
    kk: position of the extra variable
    """
    xx1 = torch.cat((x[:kk], x[-2:-1], x[kk:nParticles-1]))
    xx2 = torch.cat((x[:kk], x[-1:], x[kk:nParticles-1]))
    t1 = Norm(wf, xx1)
    t2 = Norm(wf, xx2)
    return t1, t2


def do_entangled_std(pinp, kk, seed=None, N=None):
    if N is not None:
        N_Int_Points_loc = N
    else:
        N_Int_Points_loc = N_Int_Points
    N_Int_Points_loc = int(np.sqrt(N_Int_Points_loc))
    IntElectron = [[-calc_int_electron(ppp), calc_int_electron(ppp)]]
    IntNuclei = [[-calc_int_nuclei(ppp), calc_int_nuclei(ppp)]]
    intdomain = IntElectron*nElectrons+IntNuclei*nNuclei
    intdomain_d = intdomain[:kk] + intdomain[kk+1:]
    stds = []
    means = []
    densities = []
    for _ in range(N_Int_Points_loc):
        rand_tensor = torch.rand(N_Int_Points_loc, len(intdomain_d)) * (torch.te
        rand_same = torch.rand(1, 2) * (torch.tensor(intdomain[kk])[1] - torch.t
        rand_tensor = torch.cat((rand_tensor, rand_same.repeat((N_Int_Points_loc
        t1, t2 = vmap(lambda y: get_ratio(lambda x: testwf(ppp, x), y, kk), chun
        # res = torch.log(t1 / (t2+1E-10))
        # stds.append(res.std())
        # means.append(res.mean())
        # print('mean', res.mean(), 'std', res.std())
        res = torch.log(t1 / (t2+1E-10))
        dens_mean = (t1+t2).sum()
        res_mean = (res * (t1+t2)).sum() / dens_mean
        res_std = (res**2 * (t1+t2)).sum() / dens_mean - res_mean**2
        stds.append(res_std)
        means.append(res_mean)
        densities.append(dens_mean)
        # print('mean', res_mean, 'std', res_std)

    # res_mean = np.array(means).mean()
    # res = np.array(stds).mean()
```

```
res_mean = (np.array(means)*np.array(densities)).sum() / (np.array(densities
res = (np.array(stds)*np.array(densities)).sum() / (np.array(densities)).sum
print('log of ratio full mean', res_mean, 'std', res)
return res, res_mean
```

1 The following were first ideas, but do not work

Taking a two particle wave function in one dimension $\psi(x, y)$ we suggest an entanglement measure. The wave function is not entangled, if it can be written in the form

$$\psi_{unentangled}(x, y) = \phi_1(x)\phi_2(y) \ .$$

The spatial entanglement measure should be 0 in this case. To keep it simple we will restrict the wave function to real functions. It should be straight forward to handle complex wave functions.

We construct a unentangled part of the density of states from the general two particle wave function $\psi(x, y)$.

$$\rho_{unentangle,part}(x, y) = \int_{-\infty}^{\infty} \psi^2(x, y)dx \cdot \int_{-\infty}^{\infty} \psi^2(x, y)dy \ .$$

This unentangled part of the density of states can now be subtracted from the two particle density of states of the original function

$$\rho_{entangled,part}(x, y) = \psi^2(x, y) - \rho_{unentangled,part}(x, y) \ .$$

This can be positive or negative and we suggest the entanglement measure as the integral of its absolute value

$$SEM = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |\rho_{entangled,part}(x, y)| \, dx \, dy \ .$$

We take two examples to check, if the SEM fulfills basic properties. It should be zero for a normalized unentangled wave function

$$\psi(x, y) = \sqrt{\frac{2}{\pi}} e^{-x^2} e^{-y^2} \ .$$

This results in a SEM value of 0.

Taking a fully entangled wave function

$$\psi(x, y) = \sqrt{\frac{1}{\pi}} \left(e^{-(x-2)^2} e^{-(y+2)^2} + e^{-(y-2)^2} e^{-(x+2)^2} \right) \ .$$

This results in a SEM value of 1.

We conclude, that the measure might be a possible entanglement measurement.