Configuration Model Toolkit

Carl Ganter

June 17, 2019

Contents

1 Introduction

Extended phase graphs (EPG) are indispensable for the qualitative and quantitative description of echo generation in presence of gradients. In the most common interpretation [1], the associated EPG states are motivated by non-local dephasing patterns and formally defined via voxel-scale Fourier integrals. This approach, however, impedes the proper inclusion of susceptibility effects and the quantification of echo pathways in the reconstructed signal.

We present the closely related (albeit lesser known) configuration model (CM), which does not suffer from these limitations. Relying on a microscopic representation of phase graphs, the Bloch(-Torrey) equations are solved and the transition to the voxel scale is postponed. We present a multi-dimensional CM variant, which applies to arbitrary sequences.

`CoMoTk` is a Matlab class, which implements the configuration model as described in section 2.

The transition to the voxel scale is discussed in section 3.

2 Microscopic Scale

2.1 Configuration Model

We consider an arbitrary (periodic or non-periodic) sequence[1] of instantaneous RF pulses[2] $(\alpha_\nu(\boldsymbol{x}), \varphi_\nu(\boldsymbol{x}))$, separated by time intervals of duration $\tau_{\mu(\nu)}$ during which time–dependent gradients $\boldsymbol{G}_\nu(t)$ may be played out. The corresponding time–dependent zero–order gradient moment vector $\boldsymbol{p}_\nu(t)$ is defined as usual

$$\boldsymbol{p}_\nu(t) := \gamma \int_0^t d\tau\, \boldsymbol{G}_\nu(\tau) \qquad\qquad \boldsymbol{p}_\nu\left(\tau_{\mu(\nu)}\right) =: \boldsymbol{p}_{\mu(\nu)} \tag{1}$$

If d denotes the number of different pairs[3] $\left(\tau_{\mu(\nu)}, \boldsymbol{p}_{\mu(\nu)}\right)$ in the sequence, the function

$$\mu : \mathbb{N} \to \mathbb{N} : \nu \mapsto \mu(\nu) \in \{1, \ldots, d\} \tag{2}$$

assigns a unique identifier to them. Phase encoding gradients are a typical example, when different $\boldsymbol{G}_\nu(t)$ have the same zero–order net gradient moment $\boldsymbol{p}_{\mu(\nu)}$. More generally, all $\boldsymbol{G}_\nu(t)$ with the same $\mu(\nu)$ differ by a balanced gradient.

A *static* spin, located at $\boldsymbol{x}$ with local off-resonance frequency $\omega(\boldsymbol{x})$, will accumulate the phase

$$\vartheta_\nu(\boldsymbol{x}, t) := \omega(\boldsymbol{x})\, t - \boldsymbol{p}_\nu(t)\, \boldsymbol{x} \qquad\qquad \vartheta_\nu\left(\boldsymbol{x}, \tau_{\mu(\nu)}\right) =: \vartheta_{\mu(\nu)}(\boldsymbol{x}) \tag{3}$$

for $0 \le t \le \tau_{\mu(\nu)}$.

An example of a non–periodic sequence is shown in Figure 1.

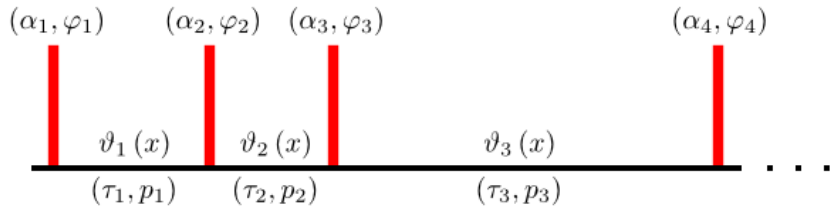

Figure 1: A non–periodic sequence

Within the configuration model of dimension d, we write the magnetization vector density $\boldsymbol{m}(\boldsymbol{x})$, immediately after any RF pulse or time interval, in the form[4]

$$\boldsymbol{m} =: \sum_{\boldsymbol{n} \in \mathbb{Z}^d} e^{i\boldsymbol{n}\vartheta}\, \boldsymbol{m}^{(\boldsymbol{n})} = \sum_{\boldsymbol{n} \in \mathbb{Z}^d} e^{i(\omega \tau_{\boldsymbol{n}} - \boldsymbol{p}_{\boldsymbol{n}} \boldsymbol{x})}\, \boldsymbol{m}^{(\boldsymbol{n})} \tag{4}$$

with

[1] We use a running index $\nu = 1, 2, \ldots$

[2] A dependence on $\boldsymbol{x}$ could be caused by B_1^+ variations. Its relevance will be situational.

[3] $\left(\tau_{\mu(\nu_1)}, \boldsymbol{p}_{\mu(\nu_1)}\right) = \left(\tau_{\mu(\nu_2)}, \boldsymbol{p}_{\mu(\nu_2)}\right) \Leftrightarrow \tau_{\mu(\nu_1)} = \tau_{\mu(\nu_2)} \wedge \boldsymbol{p}_{\mu(\nu_1)} = \boldsymbol{p}_{\mu(\nu_2)}$

[4] The d vector elements $\vartheta_\mu(\boldsymbol{x})$ are defined by Eq. (3). Keeping in mind that the configuration model is microscopic in nature, such that essentially everything possibly depends on the position $\boldsymbol{x}$, we will simplify the notation by dropping the argument $\boldsymbol{x}$, wherever it is not explicitly relevant.

$$\tau_{\boldsymbol{n}} \;:=\; \sum_{\eta=1}^{d} n_\eta\, \tau_\eta \qquad \text{and} \qquad \boldsymbol{p}_{\boldsymbol{n}} \;:=\; \sum_{\eta=1}^{d} n_\eta\, \boldsymbol{p}_\eta \tag{5}$$

We will refer to $\boldsymbol{m}^{(\boldsymbol{n})}(\boldsymbol{x})$ as *configuration vector* and $\boldsymbol{n}$ as *configuration order*.

The configuration model does not rely on a specific convention for the magnetization vector. We will, however, also derive concrete expressions for actual calculations and therefore define the complex magnetization vector (density) via[5]

$$\boldsymbol{m} \;:=\; \begin{pmatrix} (m_x + i\, m_y)/\sqrt{2} \\ m_z \\ (m_x - i\, m_y)/\sqrt{2} \end{pmatrix} \;=:\; \begin{pmatrix} m_1 \\ m_0 \\ m_{-1} \end{pmatrix} \tag{6}$$

It is related to the real magnetization density $\boldsymbol{m}_r$ by a unitary transformation $\boldsymbol{m} = \boldsymbol{U}\boldsymbol{m}_r$ with

$$\boldsymbol{U} \;:=\; \begin{pmatrix} 1/\sqrt{2} & i/\sqrt{2} & 0 \\ 0 & 0 & 1 \\ 1/\sqrt{2} & -i/\sqrt{2} & 0 \end{pmatrix} \tag{7}$$

A few comments on this arbitrary (and biased) choice:

- Dividing the transverse part by $\sqrt{2}$ (unlike in the common definition) prevents the magnetization from changing its magnitude under rotations like (9) below.
- The reordering and reindexing adapts to the intrinsic symmetries and simplifies the notation considerably.

2.2 Instantaneous RF pulses

We describe RF pulses as instantaneous rotations[6]

$$\boldsymbol{R}(\alpha_\nu, \varphi_\nu) \;:=\; \boldsymbol{R}_z(\varphi_\nu)\, \boldsymbol{R}_x(\alpha_\nu)\, \boldsymbol{R}_z(-\varphi_\nu) \tag{8}$$

applied to the local magnetization density

$$\boldsymbol{m}_+ \;=\; \boldsymbol{R}(\alpha_\nu, \varphi_\nu)\, \boldsymbol{m}_- \tag{9}$$

This operation is trivially compatible with our ansatz (4), if we define

$$\boldsymbol{m}_+^{(\boldsymbol{n})} \;:=\; \boldsymbol{R}(\alpha_\nu, \varphi_\nu)\, \boldsymbol{m}_-^{(\boldsymbol{n})} \tag{10}$$

In our chosen convention (6), we explicitly obtain the unitary matrices[7]

$$\boldsymbol{R}_x(\alpha) \;=\; \begin{pmatrix} (1+c_\alpha)/2 & -i\, s_\alpha/\sqrt{2} & (1-c_\alpha)/2 \\ -i\, s_\alpha/\sqrt{2} & c_\alpha & i\, s_\alpha/\sqrt{2} \\ (1-c_\alpha)/2 & i\, s_\alpha/\sqrt{2} & (1+c_\alpha)/2 \end{pmatrix} \tag{11}$$

[5] CoMoTk uses this convention as well.

[6] To simplify the notation, we drop the possible dependence of flip angle and phase on the position. It can be added, if necessary.

[7] Here and in the following we use the abbreviations $c_\beta := \cos(\beta)$ and $s_\beta := \sin(\beta)$.

and

$$\boldsymbol{R}_z\,(\varphi)\;=\;\begin{pmatrix} e^{i\varphi} & & \\ & 1 & \\ & & e^{-\,i\varphi} \end{pmatrix} \tag{12}$$

2.3 Time intervals

We distinguish cases without and with diffusion effects. For better readability, we use the abbreviation

$$\mu\;:=\;\mu\,(\nu) \tag{13}$$

throughout this section.

Without diffusion

In this case, we have to solve the Bloch equations

$$\frac{\partial}{\partial t}\,\boldsymbol{m}\;=\;\left[\,i\,\frac{\partial\vartheta_\nu}{\partial t}\,\boldsymbol{P}_{xy}\;-\;\boldsymbol{T}^{-1}\right]\boldsymbol{m}\;+\;\boldsymbol{T}^{-1}\boldsymbol{m}_{eq} \tag{14}$$

with ϑ_ν defined as in Eq. (3), the proton density $\boldsymbol{m}_{eq}=m_{eq}\cdot\boldsymbol{e}_z$ and the definitions[8]

$$\boldsymbol{T}\;:=\;\begin{pmatrix} T_2 & & \\ & T_1 & \\ & & T_2 \end{pmatrix}\qquad\qquad \boldsymbol{P}_{xy}\;:=\;\begin{pmatrix} 1 & & \\ & 0 & \\ & & -\,1 \end{pmatrix} \tag{15}$$

Since $\boldsymbol{P}_{xy}$ and $\boldsymbol{T}^{-1}$ commute, the formal solution can be written as

$$\boldsymbol{m}\,(t)\;=\;e^{\,i\vartheta_\nu(t)\boldsymbol{P}_{xy}\,-\,t\,\boldsymbol{T}^{-1}}\,\boldsymbol{m}\,(0)\;+\;\left(\boldsymbol{I}_3\;-\;e^{-\,t\,\boldsymbol{T}^{-1}}\right)\boldsymbol{m}_{eq} \tag{16}$$

We now insert the ansatz (4) for $\boldsymbol{m}_-:=\boldsymbol{m}\,(0)$ and get[9]

$$\boldsymbol{m}\,(t)\;=\;e^{\,i\vartheta_\nu(t)\boldsymbol{P}_{xy}}\sum_{\boldsymbol{n}\in\mathbb{Z}^d}\;e^{\,i\boldsymbol{n}\boldsymbol{\vartheta}}\,\boldsymbol{E}\,(t)\;\boldsymbol{m}_-^{(\boldsymbol{n})}\;+\;\Big(1\;-\;E_1\,(t)\Big)\,\boldsymbol{m}_{eq} \tag{17}$$

for $0\le t\le\tau_\mu$. We introduced the common notation

$$\boldsymbol{E}\,(t)\;:=\;e^{-\,t\,\boldsymbol{T}^{-1}}\;=:\;\begin{pmatrix} E_2\,(t) & & \\ & E_1\,(t) & \\ & & E_2\,(t) \end{pmatrix} \tag{18}$$

At the end of the interval, $\boldsymbol{m}_+:=\boldsymbol{m}\,(\tau_\mu)$ becomes compatible with the configuration model (4), since we then have $\vartheta_\nu\,(\tau_\mu)=\vartheta_\mu$, cf. Eq. (3).

We equate terms with equal[10] powers $e^{\,i\boldsymbol{n}\boldsymbol{\vartheta}}$ and directly obtain the recursion for the ν^{th} interval

[8] The specific form of the matrices depends on the notation. Here, we rely on Eq. (6).

[9] We could also write $\boldsymbol{R}_z\,(\vartheta_\mu)$ instead of $e^{\,i\,\vartheta_\mu\boldsymbol{P}_{xy}}$, but the latter form is better suited to equate powers in the derivation of the recursion (19).

[10] Since the equality must hold for arbitrary $\boldsymbol{x}$.

$$\boldsymbol{m}_{+}^{(\boldsymbol{n})} \;=\; \sum_{j=-1}^{1} \boldsymbol{E}^{(j)}\left(\tau_{\mu}\right) \boldsymbol{m}_{-}^{(\boldsymbol{n}-j\boldsymbol{e}_{\mu})} \;+\; \delta_{\boldsymbol{n}\boldsymbol{0}} \cdot \Big(1 \;-\; E_1\left(\tau_{\mu}\right)\Big) \boldsymbol{m}_{eq} \tag{19}$$

The longitudinal repolarization term $\propto \boldsymbol{m}_{eq}$ does not depend on $\boldsymbol{\vartheta}$ and therefore corresponds to $\boldsymbol{n} = \boldsymbol{0}$.

The matrix $\boldsymbol{E}^{(j)}$ is obtained from $\boldsymbol{E}$ by setting everything but the j^{th} column[11] to zero. $\boldsymbol{e}_{\mu} \in \mathbb{Z}^d$ is defined by $\left(\boldsymbol{e}_{\mu}\right)_j = \delta_{j\mu}$.

With diffusion

We use the Bloch–Torrey equations

$$\frac{\partial}{\partial t} \boldsymbol{m}\left(t\right) \;=\; \left[i \, \frac{\partial \, \vartheta_{\nu}}{\partial t}\left(t\right) \, \boldsymbol{P}_{xy} \;-\; \boldsymbol{T}^{-1} \;+\; \nabla \boldsymbol{D} \nabla \right] \boldsymbol{m}\left(t\right) \;+\; \boldsymbol{T}^{-1} \boldsymbol{m}_{eq} \tag{20}$$

where $\boldsymbol{D}$ denotes the diffusion tensor[12].

To solve the Bloch–Torrey equations (20), we try a generalization of the solution (17) of the Bloch equations

$$\boldsymbol{m}\left(t\right) \;=\; e^{i\vartheta_{\nu}(t)\boldsymbol{P}_{xy}} \sum_{\boldsymbol{n}\in\mathbb{Z}^d} e^{i\boldsymbol{n}\boldsymbol{\vartheta}} \, \boldsymbol{F}_{\nu}^{(\boldsymbol{n})}\left(t\right) \, \boldsymbol{E}\left(t\right) \, \boldsymbol{m}_{-}^{(\boldsymbol{n})} \;+\; \Big(1 \;-\; E_1\left(t\right)\Big) \boldsymbol{m}_{eq} \tag{21}$$

under the assumption that the only rapid spatial variations on the right hand side are caused by the gradients in $\vartheta_{\nu}\left(t\right)$

$$\nabla \, \vartheta_{\nu}\left(t\right) \;\approx\; -\, \boldsymbol{p}_{\nu}\left(t\right) \tag{22}$$

such that we only have to consider[13]

$$\begin{aligned} &\nabla \boldsymbol{D} \nabla \, e^{i\vartheta_{\nu}(t)\boldsymbol{P}_{xy}} \, e^{i\boldsymbol{n}\boldsymbol{\vartheta}} \;\approx \\ &\quad -\left[\boldsymbol{p}_{\boldsymbol{n}}^{T} \boldsymbol{D} \, \boldsymbol{p}_{\boldsymbol{n}} \cdot \boldsymbol{I}_3 \;+\; 2 \, \boldsymbol{p}_{\boldsymbol{n}}^{T} \boldsymbol{D} \, \boldsymbol{p}_{\nu}\left(t\right) \cdot \boldsymbol{P}_{xy} \;+\; \boldsymbol{p}_{\nu}^{T}\left(t\right) \boldsymbol{D} \, \boldsymbol{p}_{\nu}\left(t\right) \cdot \boldsymbol{P}_{xy}^{2} \right] e^{i\vartheta_{\nu}(t)\boldsymbol{P}_{xy}} \, e^{i\boldsymbol{n}\boldsymbol{\vartheta}} \end{aligned} \tag{23}$$

We insert (21) into the Bloch–Torrey equations (20) and obtain a differential equation for the $\boldsymbol{F}_{\nu}^{(\boldsymbol{n})}\left(t\right)$:

$$\frac{\partial}{\partial t} \boldsymbol{F}_{\nu}^{(\boldsymbol{n})}\left(t\right) \;=\; -\left[\boldsymbol{p}_{\boldsymbol{n}}^{T} \boldsymbol{D} \, \boldsymbol{p}_{\boldsymbol{n}} \cdot \boldsymbol{I}_3 \;+\; 2 \, \boldsymbol{p}_{\boldsymbol{n}}^{T} \boldsymbol{D} \, \boldsymbol{p}_{\nu}\left(t\right) \cdot \boldsymbol{P}_{xy} \;+\; \boldsymbol{p}_{\nu}^{T}\left(t\right) \boldsymbol{D} \, \boldsymbol{p}_{\nu}\left(t\right) \cdot \boldsymbol{P}_{xy}^{2} \right] \boldsymbol{F}_{\nu}^{(\boldsymbol{n})}\left(t\right) \tag{24}$$

The solution, which satisfies the boundary conditions $\boldsymbol{F}_{\nu}^{(\boldsymbol{n})}\left(0\right) \equiv \boldsymbol{I}_3$, reads

$$\boldsymbol{F}_{\nu}^{(\boldsymbol{n})}\left(t\right) \;=\; e^{-\,t\left[\boldsymbol{p}_{\boldsymbol{n}}^{T} \boldsymbol{D} \, \boldsymbol{p}_{\boldsymbol{n}} \cdot \boldsymbol{I}_3 \,+\, \boldsymbol{p}_{\boldsymbol{n}}^{T} \boldsymbol{D} \, \boldsymbol{s}_{\nu}(t) \cdot \boldsymbol{P}_{xy} \,+\, \mathrm{tr}(\boldsymbol{D} \, \boldsymbol{S}_{\nu}(t)) \cdot \boldsymbol{P}_{xy}^{2} \right]} \tag{25}$$

with

[11] Now we benefit from the unconventional indexing, introduced in our convention (6).

[12] For isotropic diffusion, we have $\boldsymbol{D} = D \cdot \boldsymbol{I}_3$. Neglecting the spatial derivative of $\boldsymbol{D}$ is consistent with Eq. (22), where we assume that gradients dominate all other spatial variations.

[13] Notation: $a := \left\| \boldsymbol{a} \right\|_2$

$$\boldsymbol{s}_\nu\,(t)\;:=\;2\,t^{-1}\int_0^t d\tau\,\boldsymbol{p}_\nu\,(\tau)\qquad\qquad\boldsymbol{S}_\nu\,(t)\;:=\;t^{-1}\int_0^t d\tau\;\boldsymbol{p}_\nu\,(\tau)\,\boldsymbol{p}_\nu^T\,(\tau)\tag{26}$$

Because of (15), $\boldsymbol{F}_\nu^{(\boldsymbol{n})}\,(t)$ must be diagonal and we apply our indexing convention (6) to its elements

$$\boldsymbol{F}_\nu^{(\boldsymbol{n})}\,(t)\;=:\;\begin{pmatrix}F_{\nu,1}^{(\boldsymbol{n})}\,(t)&&\\&F_{\nu,0}^{(\boldsymbol{n})}\,(t)&\\&&F_{\nu,-1}^{(\boldsymbol{n})}\,(t)\end{pmatrix}\tag{27}$$

We conclude that diffusion effects modify the recursion (19) by additional damping factors

$$\boldsymbol{m}_+^{(\boldsymbol{n})}\;=\;\sum_{j=-1}^{1}F_{\nu,j}^{(\boldsymbol{n}-j\boldsymbol{e}_\mu)}\,(\tau_\mu)\;\boldsymbol{E}^{(j)}\,(\tau_\mu)\;\boldsymbol{m}_-^{(\boldsymbol{n}-j\boldsymbol{e}_\mu)}\;+\;\delta_{\boldsymbol{n}\boldsymbol{0}}\cdot\Big(1\;-\;E_1\,(\tau_\mu)\Big)\,\boldsymbol{m}_{eq}\tag{28}$$

which depend on the configuration order $\boldsymbol{n}$:[14]

$$F_{\nu,j}^{(\boldsymbol{n})}\,(\tau_\mu)\;=\;e^{-\,\tau_\mu\left[\,\boldsymbol{p}_{\boldsymbol{n}}^T\boldsymbol{D}\,\boldsymbol{p}_{\boldsymbol{n}}\,+\,j\boldsymbol{p}_{\boldsymbol{n}}^T\boldsymbol{D}\,\boldsymbol{s}_\nu\,+\,j^2\,\mathrm{tr}(\boldsymbol{D}\,\boldsymbol{S}_\nu)\,\right]}\tag{29}$$

The integrals in (29) depend on the shape of $\boldsymbol{G}_\nu\,(t)$. For the sake of simplicity[15], a constant gradient is sometimes assumed, for which the moment depends linearly on time

$$\boldsymbol{p}_\nu\,(t)\;\equiv\;(t/\tau_\mu)\cdot\boldsymbol{p}_\mu\tag{30}$$

and the damping matrix (29) becomes

$$F_j^{(\boldsymbol{n})}\,(\tau_\mu)\;=\;e^{-\,\tau_\mu\left[\,\boldsymbol{p}_{\boldsymbol{n}}^T\boldsymbol{D}\,\boldsymbol{p}_{\boldsymbol{n}}\,+\,j\boldsymbol{p}_{\boldsymbol{n}}^T\boldsymbol{D}\,\boldsymbol{p}_\mu\,+\,j^2\boldsymbol{p}_\mu^T\boldsymbol{D}\,\boldsymbol{p}_\mu/3\,\right]}\tag{31}$$

Example: Stejskal-Tanner gradients

To familiarize with the formalism, we investigate an idealized spin echo sequence with $\alpha_1 = 90°$ and $\alpha_2 = 180°$. The gradient moment $\boldsymbol{p}_\nu \equiv \boldsymbol{p}$ is assumed to be constant in the two intervals of duration $\tau := \mathrm{TE}/2$, which implies that the configuration model has dimension $d = 1$.

For the signal at the spin echo, we need to combine the following facts:

- After the first excitation, only the zero order configuration $\boldsymbol{m}^{(0)}$ is occupied.
- In presence of unbalanced gradients, only the transverse[16] zero order configuration $m_1^{(0)}$ contributes to the echo after the second time interval, cf. section 3.2.
- A perfect 180° pulse swaps the transverse vector components $j = 1$ and $j = -1$.

[14] $\boldsymbol{s}_\nu := \boldsymbol{s}_\nu\,(\tau_\mu)$, $\boldsymbol{S}_\nu := \boldsymbol{S}_\nu\,(\tau_\mu)$ and $\mathrm{tr}\,()$ denotes the trace.

[15] For large $\boldsymbol{n}$, the quadratic term, which does not depend on the gradient shape, can become dominant. The assumption (30), for which the steady–state of unbalanced SSFP has been solved analytically in [2], can be acceptable at $\mathrm{TR} \ll T_2$, for example.

[16] The index "1" does not refer to the running index ν but to the transverse vector component, cf. Eq. (6).

The spin echo is thus generated by the train

$$m_0^{(0)} \;\overset{90^\circ}{\longrightarrow}\; m_{-1}^{(0)} \;\overset{\tau}{\longrightarrow}\; m_{-1}^{(-1)} \;\overset{180^\circ}{\longrightarrow}\; m_1^{(-1)} \;\overset{\tau}{\longrightarrow}\; m_1^{(0)} \tag{32}$$

and, in view of the recursion (28), the diffusion related damping factor is given by $F_{2,1}^{(-1)} \cdot F_{1,-1}^{(0)}$. For isotropic diffusion, we explicitly get

$$F_{2,1}^{(-1)} \cdot F_{1,-1}^{(0)} \;=\; e^{-\,\tau D\left[\,p^2\,-\,\boldsymbol{p}^T\boldsymbol{s}_2\,+\,\mathrm{tr}(\boldsymbol{S}_2)\right]} \cdot e^{-\,\tau D\;\mathrm{tr}(\boldsymbol{S}_1)} \;=:\; e^{-\,b\cdot D} \tag{33}$$

We define the composed gradient $\boldsymbol{G}(t)$ by $\boldsymbol{G} = \boldsymbol{G}_1$ on $[0,\tau]$ and[17] $\boldsymbol{G} = -\,\boldsymbol{G}_2$ on $[\tau, 2\tau]$ and obtain the result

$$b \;=\; \tau\left[\,p^2 \;-\; \boldsymbol{p}^T\boldsymbol{s}_2 \;+\; \mathrm{tr}\,(\boldsymbol{S}_2) \;+\; \mathrm{tr}\,(\boldsymbol{S}_1)\,\right] \;=\; \gamma^2 \int_0^{2\tau} dt \left[\,\int_0^t dt'\,\boldsymbol{G}\,(t')\right]^2 \tag{34}$$

in agreement with the literature [3]. For the specific case, when $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$ are of identical, rectangular shape[18], we get

$$b \;=\; p^2 \cdot \left(\Delta - \frac{\delta}{3}\right) \tag{35}$$

where δ and Δ denote the width and relative displacement of the two gradients, respectively.

3 Voxel Scale

At first glance, the configuration model (4) is just a cumbersome and redundant way to rewrite the microscopic magnetization density $\boldsymbol{m}\,(\boldsymbol{x})$.

The situation changes beyond the microscopic scale, in particular for the reconstructed *voxel* m_ρ, when its true value becomes apparent.

Actually, two fundamental mechanisms are available for signal localization:

- Selective Excitation
- Spatial Encoding

Both rely on the application of gradients, effecting more or less complicated signal modulations in the vininity of a given location $\boldsymbol{x}$. Under the assumption that other causes[19] for these variations can be neglected on the voxel scale, the superposition of configurations in Eq. (4) encodes just this information via the phase factors $e^{i\boldsymbol{n}\boldsymbol{\vartheta}(\boldsymbol{x})}$.

As an example, how to apply this knowledge for both localization mechanisms, let us consider some 2D sequence with Cartesian sampling and slice selective excitation. The finite readout duration is neglected and we assume the coordinate axes to be aligned parallel to the reconstructed image, such that $x_{1,2}$ encode the locations along the orthonormal in-plane vectors $\boldsymbol{e}_{1,2}$ and x_3 the position along the slice normal $\boldsymbol{e}_3$.

[17] The different sign is required because of the 180° pulse, cf. the discussion in [3].
[18] Specifically, we assume directed gradients of the form $\boldsymbol{G}_\nu\,(t) = G_\nu\,(t)\cdot\boldsymbol{p}/p$.
[19] Tissue properties, partial volume effects, B_0 inhomogeneity (beyond suceptibility variations), to name a few.

3.1 Spatial encoding

Due to finite sampling, the discrete reconstructed voxel signal m_ρ at position $\boldsymbol{x}_\rho$ results from convolution of the transverse magnetization density $m(\boldsymbol{x})$ with a *point spread function* $\phi(\boldsymbol{x})$

$$m_\rho \;\propto\; \phi * m(\boldsymbol{x}_\rho) \tag{36}$$

which in our example is just a scaled sinc function[20]

$$\phi(\boldsymbol{x}) \;:=\; \prod_{j=1}^{2} \operatorname{sinc}\left(\frac{x_j}{\Delta x_j}\right) \tag{37}$$

where Δx_j denotes the pixel resolution in direction j.

We now insert the configuration model (4) into (36) with $f^{(\boldsymbol{n})} := e^{i\,\omega\tau_{\boldsymbol{n}}}\, m^{(\boldsymbol{n})}$ and get after short calculation

$$m_\rho \;\propto\; \sum_{\boldsymbol{n}\in\mathbb{Z}^d} \int d\boldsymbol{k}\, e^{i\,\boldsymbol{k}\boldsymbol{x}_\rho}\, \hat{f}^{(\boldsymbol{n})}(\boldsymbol{k}) \cdot \prod_{j=1}^{2} u\left(\frac{\pi}{\Delta x_j} - |k_j + p_{\boldsymbol{n},j}|\right) \tag{38}$$

where u is the unit step function.

If the support of $\hat{f}^{(\boldsymbol{n})}$ in direction j is approximately[21] bounded by $\pi/\Delta x_j$, only configurations with

$$|\,p_{\boldsymbol{n},j}| \;<\; \frac{2\pi}{\Delta x_j} \tag{39}$$

contribute to m_ρ.

Essentially, this matches the often encountered statement that crusher gradients should effect a 2π dephasing over the voxel dimension, in order to be effective. But we also see that this is only approximately true and depends on the severity of partial volume effects.

To summarize, $\boldsymbol{p}_{\mathbf{n}}$ provides information, which configurations $m^{(\boldsymbol{n})}$ contribute to the reconstructed signal m_ρ.

3.2 Selective excitation

In real acquisitions, selective excitation is performed via application of a resonant $\boldsymbol{B}_1^+$ field in presence of slice encoding gradients. For example, an amplitude modulated RF pulse is usually split into a few hundred intervals of equal duration, within each of which the gradient and $\boldsymbol{B}_1^+$ are held constant[22]. Since rotations around different axes do not commute in general, RF pulse design treats RF pulses as a series of instantaneous small tip angle pulses interleaved by short intervals[23] τ of constant gradient moments $\boldsymbol{p}$ in direction of the slice normal $\boldsymbol{e}_3$, cf. [4]. The accuracy of this approximate description is determined by the choice of τ.

[20] We assume the normalized version: $\operatorname{sinc}(x) := \sin \pi x/\pi x$

[21] This means that the reconstructed resolution recovers the essential fine structure of the sample. In a strict sense, $\hat{f}^{(\boldsymbol{n})}$ is not bouded at all, as it is the Fourier transform of a bounded function.

[22] VERSE pulses are an exception.

[23] In the configuration model, the small intervals generate a separate dimension with a unique index, say, $\mu = 1$ and the assignments $\tau_1 = \tau$ and $\boldsymbol{p}_1 = \boldsymbol{p}$.

For the reconstructed voxel m_ρ, Eq. (36) still applies in principle, since it also includes an unbounded integral along the slice normal $\boldsymbol{e}_3$. Due to the approximate description of the RF pulse, however, the configuration model according to Eq. (4) becomes periodic in this direction

$$m(\boldsymbol{x}) \;\equiv\; m\left(\boldsymbol{x} + n \cdot \frac{2\pi}{p} \cdot \boldsymbol{e}_3\right) \qquad\qquad n \in \mathbb{Z} \tag{40}$$

and the x_3 integral should therefore be restricted to the interval $[-\pi/p, \pi/p]$. In presence of several pulses with possibly different p_μ, the *excitation* pulse[24] defines the integration interval. In general, $\boldsymbol{p}$ is not the only gradient mit a nonzero component in direction of $\boldsymbol{e}_3$. For example, to restore spin coherence across the slice, subsequent time intervals include a rephasing moment parallel to $\boldsymbol{e}_3$.

The effect of selective excitation can therefore be handled by a weighting factor $w_{\boldsymbol{n}}$ in Eq. (4)

$$e^{-\,i\,\boldsymbol{p}_{\boldsymbol{n}}\boldsymbol{x}} \cdot m^{(\boldsymbol{n})} \;\rightarrow\; w_{\boldsymbol{n}} \cdot m^{(\boldsymbol{n})} \tag{41}$$

where

$$w_{\boldsymbol{n}} \;:=\; \frac{p}{2\pi} \cdot \int_{-\pi/p}^{\pi/p} dx_3\, e^{-\,i\,p_{\boldsymbol{n},3}x_3} \;=\; \operatorname{sinc}\left(\frac{p_{\boldsymbol{n},3}}{p}\right) \tag{42}$$

Since p is rather small, it is sometimes possible to assume that the third component of all $\boldsymbol{p}_\mu$ is some integer multiple of p. In this case, Eq. (42) simplifies to

$$w_{\boldsymbol{n}} \;=\; \begin{cases} 1 & : \quad p_{\boldsymbol{n},3} = 0 \\ 0 & : \quad p_{\boldsymbol{n},3} \neq 0 \end{cases} \tag{43}$$

If, additionally, all nonzero in-plane gradient moments refer to (large enough) crusher gradients, the restrictions due to spatial encoding and selective excitation can be combined in a single, handy statement:

$$w_{\boldsymbol{n}} \;=\; \begin{cases} 1 & : \quad \boldsymbol{p}_{\boldsymbol{n}} = \boldsymbol{0} \\ 0 & : \quad \boldsymbol{p}_{\boldsymbol{n}} \neq \boldsymbol{0} \end{cases} \tag{44}$$

3.3 Susceptibility effects

Susceptibility related intra-voxel frequency variations $\omega_s(\boldsymbol{x})$, as part of the local resonance frequency $\omega(\boldsymbol{x})$, are typically considered as independent of gradient induced frequency modulations in the integral (36).

Under this assumption, the fluctuations are assumed to be distributed according to some zero-mean density $p(\omega_s)$ within the voxel. The associated damping factor depends on the configuration (via $\tau_{\boldsymbol{n}}$) and is given by

$$\hat{p}(\tau_{\boldsymbol{n}}) \;:=\; \int d\omega_s\, e^{i\,\omega_s\tau_{\boldsymbol{n}}} \cdot p(\omega_s) \tag{45}$$

[24] This should handle virtually all relevant cases: For (T)SE sequences, the excitation pulse is uniquely defined, since excitation due to imperfect refocusing cannot enter the signal (these paths are suppressed by the crusher gradients). On the other hand, every pulse in an SSFP sequence can excite spins, but here the RF pulses are usually identical (at least with respect to duration and gradient moment).

To calculate m_ρ, we replace $m^{(\boldsymbol{n})} \to \hat{p}\,(\tau_{\boldsymbol{n}}) \cdot m^{(\boldsymbol{n})}$ in the configuration model (4) for all contributing $\boldsymbol{n}$.

Most commonly, a Lorentzian distribution $p\,(\omega_s)$ is assumed and the damping factor $\hat{p}\,(\tau_{\boldsymbol{n}})$ takes the familiar form

$$\hat{p}\,(\tau_{\boldsymbol{n}}) \;=\; e^{-\,R_2'\tau_{\boldsymbol{n}}} \tag{46}$$

4 Design Notes

The Matlab class `CoMoTk` implements the configuration model according to its definition in section 2. In addition to executing arbitrary sequences, various first order partial derivatives (with respect sequence and tissue parameters) may be extracted as well.

4.1 Units

The toolkit assumes the following units throughout:

- **time:** [ms]
- **relaxation rate:** [1/ms]
- **angular frequency:** [rad/ms]
- **apparent diffusion coefficient:** [μm^2/ms]
- **angles:** [rad]
- **length, position:** [μm]
- **gradient moment, spatial frequency:** [1/μm]

4.2 Spoiler gradients

Spoiler and crusher gradients can be differentiated by their intended action. While the latter are used in sequences for de– *and* rephasing, the task of the former is to destroy transverse coherences reliably [3]. In principle, this goal can be (partly) reached by two effects:

- **Voxel scale:** Suppression of unwanted configurations via crusher gradients (included in $p_{\boldsymbol{n}}$), as outlined in section 3.1.
- **Microscopic scale:** By diffusion effects.

With respect to the second mechanism, it follows from Eq. (29) and the recursion (28) that ideal spoiling can be accomplished in the limit

$$\mathrm{tr}\,(\boldsymbol{D}\,\boldsymbol{S}_\nu) \;\to\; \infty \tag{47}$$

since then we have

$$\boldsymbol{F}^{(\boldsymbol{n})}\,(\tau_\mu) \;\to\; \begin{pmatrix} 0 & & \\ & e^{-\,\tau_\mu\,\boldsymbol{p}_{\boldsymbol{n}}^T\boldsymbol{D}\,\boldsymbol{p}_{\boldsymbol{n}}} & \\ & & 0 \end{pmatrix} \tag{48}$$

and transverse magnetization is eliminated in all configuration orders.

Note that the condition (47) does not necessarily[25] require $p_\mu \to \infty$ (or $p_{\boldsymbol{n}} \to \infty$). On the other hand, $p_\mu \to \infty$ will usually require[26] (47) as well.

4.3 General properties of occupied configurations

Proposition 1. *In absence of magnetization transfer and for $T_2 \leq T_1$, we always have*

$$\sum_{\boldsymbol{n}} \left\| \boldsymbol{m}^{(\boldsymbol{n})} \right\|_2^2 \;\leq\; m_{eq}^2 \tag{49}$$

Proof. The RF pulses do not affect $\left\| \boldsymbol{m}^{(\boldsymbol{n})} \right\|$. Therefore we only need to consider the time intervals and the proof can be done by induction. From (28), we derive

$$\left| m_{+,j}^{(\boldsymbol{n})} \right| \;=\; \begin{cases} \left| F_{\nu,0}^{(\boldsymbol{0})}\, E_{00}\, m_{-,0}^{(\boldsymbol{0})} + (1-E_1)\, m_{eq} \right| & : \quad \boldsymbol{n} = \boldsymbol{0} \;\wedge\; j = 0 \\ \left| F_{\nu,j}^{(\boldsymbol{n}-j\boldsymbol{e}_\mu)}\, E_{jj} \right| \left| m_{-,j}^{(\boldsymbol{n}-j\boldsymbol{e}_\mu)} \right| & : \quad \text{else} \end{cases} \tag{50}$$

In combination with $T_2 \leq T_1$ and assuming that (49) holds for $\boldsymbol{m}_-^{(\boldsymbol{n})}$, we therefore obtain

$$\begin{aligned} \sum_{\boldsymbol{n}} \left\| \boldsymbol{m}_+^{(\boldsymbol{n})} \right\|_2^2 \;&\leq\; E_1^2 \cdot \sum_{\boldsymbol{n}} \left\| \boldsymbol{m}_-^{(\boldsymbol{n})} \right\|_2^2 \;+\; 2\,E_1\,(1-E_1) \left| m_{-,0}^{(\boldsymbol{0})} \right| m_{eq} \;+\; (1-E_1)^2\, m_{eq}^2 \\ &\leq\; \left(E_1^2 \;+\; 2\,E_1\,(1-E_1) \;+\; (1-E_1)^2 \right) m_{eq}^2 \\ &=\; m_{eq}^2 \end{aligned} \tag{51}$$

which completes the proof. □

4.4 Storage considerations

The actual state of the configuration is defined by the set $\{\boldsymbol{n}\}$ of occupied configurations and their associated configuration vectors $\boldsymbol{m}^{(\boldsymbol{n})}$.

The size of the set, n_c, will increase with every time interval

$$n_c \;\to\; \begin{cases} n_c + 2d & : \quad d \to d \\ 3n_c & : \quad d \to d+1 \end{cases} \tag{52}$$

The first case applies, if the same time interval μ has been played out before[27], whereas the second case applies for new intervals[28].

Eq. (52) implies that n_c becomes particularly large for high dimensional configuration models. The most extreme scenario is a fingerprinting-type excitation pattern with distinct random intervals τ_μ. This pattern generates $n_c = 3^n$ occupied configurations after $n = d$ time intervals, easily exceeding the memory of actual computer hardware after less than about $n \approx 20$ intervals.

[25] Strong balanced gradients with $p_\mu = 0$ are a simple counterexample. In the specific case (30), however, the conditions (47) and $p_\mu \to \infty$ are equivalent.

[26] More accurately, if we replace "∞" by "large", this follows from the actual limitations of gradient hardware.

[27] In this case, the dimension of the configuration model, d remains unchanged.

[28] Here, the dimension d is increased by 1.

Fortunately, due to relaxation, magnetization forgets about its initial state with the consequence that many (if not most) of the occupied configuration vectors $\boldsymbol{m}^{(\boldsymbol{n})}$ may be safely ignored. Formally, this immediately follows from the upper bound on $\sum_{\boldsymbol{n}} \left\|\boldsymbol{m}^{(\boldsymbol{n})}\right\|_2^2$, derived in Proposition 1.

Based on some given, user-defined limit ε, we will therefore repeatedly discard all configuration vectors with $\left\|\boldsymbol{m}^{(\boldsymbol{n})}\right\|_2 < \varepsilon$. To this end, we invoke the method `meltdown()` after every time interval.

For this to work properly, the actual set $\boldsymbol{m}^{(\boldsymbol{n})}$ is stored in a two-dimensional array `m` of size[29] $3 \times n_a$, where n_a specifies the allocated (not necessarily occupied) space. This storage concept requires a considerable amount of bookkeeping, which is described in more detail in section 4.5.

To improve performance further, reallocations[30] and redundant computations[31] are minimized as well.

4.5 Bookkeeping

The `n_conf` *stored* configuration orders $\boldsymbol{n}$ of dimension `d` are collected in the $n_a \times \mu_a$ array `n`. The associated location in the allocated storage is determined by the logical $n_a \times 1$ array `b_n` and the $1 \times \mu_a$ array `b_mu`.[32]

The `d` distinct dimensions μ are stored in the $1 \times \mu_a$ array `mu`.

Prior to and after the ν^{th} time interval, we denote the set of *stored* configurations $\boldsymbol{n}$ by S_ν^- and S_ν^+, respectively. To implement the time intervals properly, particularly recursions like (28), it is crucial to guarantee that $S_\nu^\pm$ does not contain any holes: If $\boldsymbol{n} \in S_\nu^\pm$ and $\boldsymbol{n} + c \cdot \boldsymbol{e}_\mu \in S_\nu^\pm$ for some $\boldsymbol{n} \in \mathbb{Z}^d$ and $c \in \mathbb{Z} > 0$, we must have $\boldsymbol{n} + b \cdot \boldsymbol{e}_\mu \in S_\nu^\pm$ for every $1 \leq b \in \mathbb{Z} < c$.

Since RF pulses do not change the set of occupied configurations, we set $S_{\nu+1}^- = S_\nu^+$.

For each $\boldsymbol{n} \in S_\nu^-$ and each dimension μ, the information whether $\boldsymbol{n} \pm \boldsymbol{e}_\mu \in S_\nu^-$ is fulfilled, is stored in a separate variable.[33] In case of $\boldsymbol{n} \pm \boldsymbol{e}_\mu \in S_\nu^-$, the corresponding location in memory must be available as well.[34]

Due to the method `meltdown()`, the `n_conf` $\leq n_c$ *stored* configurations are some subset of $\mathbb{Z}^{\mathtt{d}}$ with[35] `d` $\leq d$. We demand that the stored set remains connected along each dimension, i.e. free of holes, as described above. We also demand that $\boldsymbol{n} = \boldsymbol{0}$ is included at all times.

5 Usage

To familiarize with the toolkit, the scripts in the `test` and `examples` folders are recommended as a good starting point. Following, a brief overview of the basic functionality.

5.1 Initialize `CoMoTk`

First, we need to create an instance of `CoMoTk`:

```
cm = CoMoTk;
```

[29] The first dimension refers to the three vector components.

[30] Mainly achieved via a sufficiently large initial value of n_a.

[31] Some quantities, which are repeatedly needed in calls of identical RF pulses or time intervals are stored for reuse. When possible, updates are limited to newly entering configurations.

[32] Obvioulsly, `sum( b_n )` and `sum( b_mu )` must be equal to `n_conf` and `d`, respectively.

[33] We define two logical $n_a \times \mu_a$ arrays `b_up_free` and `b_do_free` as follows: For each stored configuration $\boldsymbol{n}$ and dimension μ, the associated entry in `b_up_free` is `true`, if the direct neighbor $\boldsymbol{n} + \boldsymbol{e}_\mu$ is *not* stored and `false` otherwise. Similarly, `b_do_free` is `true`, if $\boldsymbol{n} - \boldsymbol{e}_\mu$ is *not* stored and `false` otherwise.

[34] Similarly, we define two $n_a \times \mu_a$ arrays `idx_up` and `idx_do` as follows: For each stored configuration $\boldsymbol{n}$ and dimension μ, with the associated entry in `b_up_free` equal to `false`, the corresponding value `idx_up` gives the index (with respect to the first dimenasion of size n_a) of the direct neighbor $\boldsymbol{n} + \boldsymbol{e}_\mu$. For the other direction, $\boldsymbol{n} - \boldsymbol{e}_\mu$, the array `idx_do` is defined in complete analogy.

[35] A populated dimension μ, for which no configuration orders $n_\mu \neq 0$ are stored, is de facto nonexistent and can be discarded. This can, for example, happen, if the associated time period τ_μ has not been played out for a long time.

Next, we have to setup a few mandatory tissue parameters. In the simpliest case of a 1-peak model without diffusion, this can look like this:

```
cm.R1 = 0.01;          % longitudinal relaxation rate
cm.R2 = 0.1;           % transverse relaxation rate
cm.D = 0;              % apparent diffusion coefficient
```

It is also possible, to specify more complex n-peak models with variable relative weighting ($\propto$ proton density), chemical shift and diffusivity. Here, a 2-peak example:

```
cm.R1 = [ 0.01, 0.02 ];
cm.R2 = [ 0.1, 0.2 ];
cm.D = [ 3, 1.5 ];
cm.w = [ 0.6, 0.4 ];      % relative weight (does not need to add to 1)
cm.dom = [ 0, -0.1 ];     % chemical shift (angular frequency)
```

Setting `w` and `dom` is optional. If not set, they default to 1 and 0, respectively.

Another optional variable is the relative B_1^+ field (default = 1):

```
cm.B1 = 0.8;
```

There are further options, for which it is possible to modify the default settings. The most important one is the desired accuracy. A nonzero value of `epsilon` is set, if the number of stored configurations needs to be restricted, e.g. due to memory or performance restrictions.

```
options = cm.options;      % get default options

options.alloc_d = 3;       % allocated number of dimensions
options.alloc_n = 10000;   % allocated number of configurations
options.epsilon = 0;       % for maximal accuracy (enough memory needed)
options.verbose = true;    % for more output
options.debug = true;      % for debugging purposes (look into CoMoTk.m)

cm.options = options;      % activate new options
```

Now we have to specify the initial configuration vector, corresponding to $\boldsymbol{n} = 0$. It is supplied as a real vector (row or column) in the usual convention (m_x, m_y, m_z):

```
cm.init_configuration ( [ 0; 0; 1 ] );  % longitudinal magnetization
```

In addition to $m_\nu^{(\boldsymbol{n})}$, knowledge of certain partial derivatives $\partial m_\nu^{(\boldsymbol{n})}/\partial\xi$ is sometimes desired as well, e.g. for numerical optimization. This is supported in `CoMoTk` for $\xi \in \{R_1, R_2, D, B_1, \alpha_\mu, \varphi_\mu, \tau_\mu, \boldsymbol{p}_\mu, \boldsymbol{s}_\mu, \boldsymbol{S}_\mu\}$. The following command (a full example, involving every possible variable) is used, to inform `CoMoTk`,

which derivatives need to be calculated:[36]

```
cm.set_derivatives ( ...
    'R1', [ 1, 3 ], ...          % tissue handles (n-peak model)
    'R2', 2, ...                 % for a 1-peak model the handle (= 1)
    'D', [ 2, 3 ], ...           % must be supplied as well
    'B1', ...                    % only B1 has no handle
    'FlipAngle', [ 2, 3 ], ...   % non-tissue handles can be chosen
    'Phase', [ 2, 1004, 12 ], ... % arbitrarily
    'tau', [ 3, 4, 6 ], ...      % time interval (= \mu)
    'p', [ 1, 2, 4 ], ...        % gradient moment (see below)
    's', [ 4, 7 ] ...            % gradient shape (see below)
    );
```

Of course, only the desired subset of parameter/handle combinations needs to be supplied. If the command is not given, no derivatives are calculated by default.

After having initialized everything, we proceed with the actual sequence. Here, we apply instantaneous RF pulses and time intervals, typically in an alternate fashion.[37]

5.2 RF pulse

Calling an RF pulse is as simple as:

```
cm.RF( flip, phase );  % RF pulse with flip angle and phase [rad]
```

If partial derivatives with respect to flip angle(s) and/or phase(s) have been set before, the associated handles can be additionally supplied according to the following format:[38]

```
cm.RF( flip, phase, 'FlipAngle', flip_handle, 'Phase', phase_handle );
```

5.3 Time interval

Whether the time derivatives are needed or not, executing the time interval always needs specification of the (otherwise arbitrary) index `mu` $:= \mu$, which is defined as in section 2. At least in the first call of each distinct interval, the duration `tau` $:= \tau_\mu$ must be supplied also:

```
cm.time( mu, 'tau', tau );
```

If we also require the gradient moment `p` $:= \boldsymbol{p}_\mu$ for the simulations (diffusion effects) or the results (e.g. slice profile), we have to add this parameter as well:[39]

[36] Currently, `CoMoTk` supports isotropic diffusion only. Therefore, the gradient shape is determined by `s` $:= (\boldsymbol{s}_\mu, \mathrm{tr}\,(\boldsymbol{S}_\mu))$. Derivatives with respect to `s` for a given interval μ make only sense, if the shape does not change.

[37] There are no restrictions, though, i.e. multiple subsequent RF pulses or time intervals are allowed as well.

[38] If only the flip angle derivative is needed, the phase handle is not required (and vice versa). The user is responsible for the validity of value/handle combinations. Otherwise, the result is unpredicted.

[39] `p` must be a 3×1 or 1×3 array. If we set an element of `p` equal to `Inf`, it is interpreted as an ideal spoiler, as defined in section 4.2.

```
cm.time( mu, 'tau', tau, 'p', p );
```

Called like this, a constant gradient shape according to (30) and (31) is assumed.

For arbitrary shapes, the variable `s` $:= (\boldsymbol{s}_\nu, \mathrm{tr}\,(\boldsymbol{S}_\nu))$ must be added too:[40]

```
cm.time( mu, 'tau', tau, 'p', p, 's', s );
```

In later calls, only the handle is required:[41]

```
cm.time( mu );
```

Note, however, that this shorthand form is not safe for nonzero `epsilon`, since the dimension `mu` could have been eliminated (together with any knowledge about `tau` and `p`) by a previous call of `meltdown()`. In case of doubt, always specify all parameters.

5.4 Get results

After any RF pulse or time interval, we can obtain the sum[42] (4) like this:

```
res = cm.sum( param );
```

`param` is a structure with optional fields:

`omega` = Local angular off-resonance frequency $\omega(\boldsymbol{x})$

`x` = Position $\boldsymbol{x}$ according to the definition (3)

`b_n` = Subset from the set of stored configuration orders, $S_\nu^\pm$, to be included in the result.

`w_n` = explicit weighting factors (`length( w_n ) = sum( b_n )`)

For unset fields, the following defaults are assumed:

- `omega` = 0
- `x` = 0
- `b_n` = `cm.b_n` (= whole sum in Eq. (4))
- `w_n` = 1

The result is returned separately as transverse[43] (`res.xy`) and longitudinal (`res.z`) component. Calculated derivatives with respect to `X` are returned as `res.dm_dX.xy` and `res.dm_dX.z`, where `X` is any member of the set {`R1`, `R2`, `D`, `B1`, `FlipAngle`, `Phase`, `tau`, `p`, `s`}. We have `size(res.dm_dX) = [a,b]`, where `a = 1`, except for `X = p` with `a = 3` and `X = s` with `a = 4`. The second dimension `b` is just the number of derivatives to be calculated for each parameter as defined in `set_derivatives()`.

For a restriction of the summation, the `find` method can be used to generate `b_n`:

[40] `s` must be a 4×1 or 1×4 array.

[41] The full version is also allowed. Of course, `s` still needs to be supplied, if the shape is variable.

[42] For the isochromat the sum over all tissues (if there is more than one) is performed as well.

[43] In the traditional interpretation `res.xy` $= m_x + i\,m_y = \sqrt{2} \cdot m_1$, where the last term corresponds to our chosen notation in Eq. (6).

```
b_n = cm.find( mu, n );
```

If `mu` and `n` are scalars, we have `b_n` $= \{\boldsymbol{n} \in S_\nu^\pm : n_{\texttt{mu}} = \texttt{n}\}$. If no matching configurations are found, `b_n = []` is returned.

`mu` and `n` can also be 1d-arrays of equal length. This can be used as a shorthand for a combination with the `|` operator. For example, we obtain for arrays of length 3 a result equivalent to

```
b_n = ...
    cm.find( mu( 1 ), n( 1 ) ) | ...
    cm.find( mu( 2 ), n( 2 ) ) | ...
    cm.find( mu( 3 ), n( 3 ) );
```

If we want to single out a specific configuration $\boldsymbol{n} \in S_\nu^\pm$, we have to use the `&` operator explicitly. For example, we get for $d = 3$:

```
b_n = ...
    cm.find( cm.mu( 1 ), n( 1 ) ) & ...
    cm.find( cm.mu( 2 ), n( 2 ) ) & ...
    cm.find( cm.mu( 3 ), n( 3 ) ) );
```

It is also possible, to restrict the mask `b_n` directly. For example, to extract all stored configurations, which satisfy $\boldsymbol{p}_{\boldsymbol{n}} = \boldsymbol{0}$, cf. Eq. (44), one could use

```
b_n = cm.b_n & reshape( ~any( cm.p_n ), size( cm.b_n ) );
```

References

[1] Matthias Weigel. Extended phase graphs: Dephasing, rf pulses, and echoes - pure and simple. *Journal of Magnetic Resonance Imaging*, 41(2):266–295, 2014.

[2] D. E. Freed, U. M. Scheven, L. J. Zielinski, P. N. Sen, and M. D. Hürlimann. Steady-state free precession experiments and exact treatment of diffusion in a uniform gradient. *The Journal of Chemical Physics*, 115(9):4249–4258, 2001.

[3] Xiaohong Joe Zhou Matt A. Bernstein, Kevin F. King. *Handbook of MRI pulse sequences*. Academic Press, 2004.

[4] John Pauly, Patrick Le Roux, Dwight Nishimura, and Albert Macovski. Parameter relations for the shinnar-le roux selective excitation pulse design algorithm (nmr imaging). *IEEE Transactions on Medical Imaging*, 10(1):53–65, 1991.